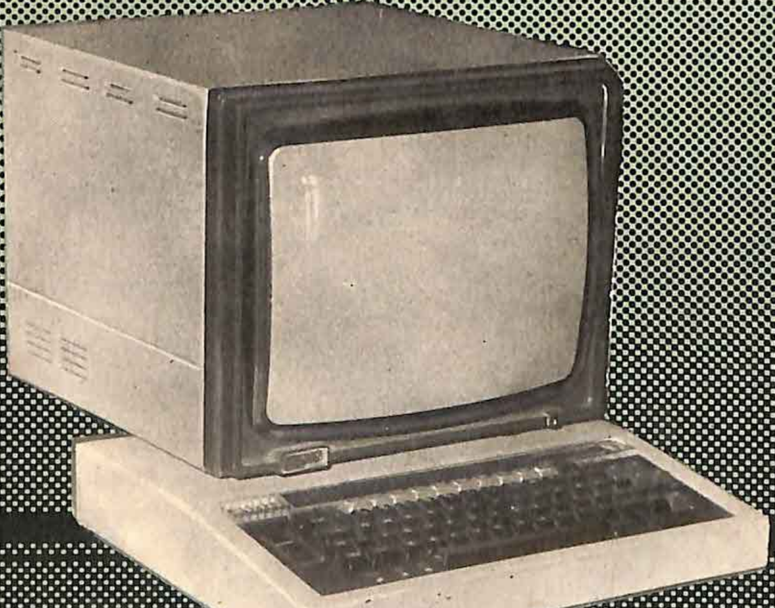
BBC BASIC
PROGRAMMING

M S SOLANKI



A USERS' MANUAL
WITH EXERCISES

BBC BASIC PROGRAMMING

A Users' Manual with Exercises

M.S. SOLANKI
B.TECH.(MECH. ENGG.), B.ED., M.SC.(MATHS)
P.G.T.(Maths) and Computer Teacher
Kendriya Vidyalaya, Bhopal

Acc No - 15330

BOMBAY
OXFORD UNIVERSITY PRESS
DELHI CALCUTTA MADRAS
1988

Oxford University Press
Oxford New York Toronto
Delhi Bombay Calcutta Madras Karachi
Petaling Jaya Singapore Hong Kong Tokyo
Nairobi Dar es Salaam
Melbourne Auckland
and associates in
Berlin Ibadan

© Oxford University Press 1988

SBN 19 562221 9

ACCNO – 15350

Printed by R.N. Kothari, Konam Printers,
Tardeo, Bombay 400 034 and published
by S.K. Mookerjee, Oxford University Press
Oxford House, Apollo Bunder, Bombay 400 039.

Foreword

With the large-scale introduction of BBC Microcomputer systems in Indian Schools, there has arisen a strong need for a book which will help students and teachers to understand computers and work with them. This book attempts to meet this need. It is specifically written for the BBC microcomputer and the version of BASIC used by it. The language of the book is lucid and the treatment simple. The development of the content is graded. The book moves with the basic concepts of computers and takes the reader gradually through the different steps in programming. Starting with an introduction to flow charting and simple commands, the book sails smoothly to colour Graphics and ~~Sound,~~ which are unique features of the BBC Microcomputer. The exercises at the end of each chapter will help the reader to test his understanding of the text. The book furnishes over 100 listings of Programs, which should serve as good examples of practical applications. A summary of commands and computer terminology provides the reader with a ready reference.

Though the book will primarily be useful to the students and teachers involved in the implementation of the CLASS project, it should prove a valuable guide for anyone interested in learning BASIC as there are only minor variations in the different versions of BASIC.

I wish the book all success.

K. VENKATASUBRAMANIAN

Professor and Head of the Deptt.,
Electrical and Electronics Engg. and
Co-ordinator, Project CLASS Resource Centre,
Technical Teachers' Training Institute,
Bhopal 462 002.

Preface

Computers have a vital role to play in today's world. Every second person needs or wants to learn how to use a computer. In line with these felt needs, in 1984 the Government of India initiated the CLASS (Computer Literacy and Studies in Schools) Project to spread computer consciousness and computer skills amongst school-students, and provided a wide cross-section of schools across the country with BBC Microcomputers.

This book has been written as a text to meet the needs of students of Classes XI and XII of schools under the CLASS PROJECT and the many other schools who have independently instituted computer classes. Although the book is written specifically for those using the BBC Microcomputer, with slight modifications it can be used with any other microcomputer which is based on BASIC computer language.

A step by step approach has been adopted in structuring the book to enable students to acquire computer skills painlessly, even without the guidance of a teacher. Great stress has also been laid on exercises, both solved and unsolved, at the end of each lesson to give students confidence in their mastery of one computer skill before proceeding to the next. A special feature is the inclusion of chapters on Advanced Graphics, Animation, and Sound and Teletext Control.

In writing the book I have been greatly aided by the standard literature on the subject and am greatly indebted to the respective authors. I am extremely grateful also to Prof. K V Subramanian for his able guidance, my principal and colleagues for their encouragement, and to my students for their assistance in the preparation of the manuscript and testing the programmes. My thanks go also to Mr Indorkar for the preliminary drawings, Mr O P Gupta for typing the text and, last but not least, Oxford University Press, my publishers, for their cooperation in editing, designing and producing the book. I need hardly add that I own full responsibility for any errors that may have inadvertently crept into the text and would greatly welcome readers' suggestions.

M S SOLANKI

Kendriya Vidyalaya
Bhopal
26 January 1988

Contents

Foreword by Prof. K. Venkatasubramanian iii
Preface v
1 Electronic Computers and their History 1
2 The Keys of the BBC Micro 12
3 Flow Chart and Algorithm 17
4 Some Useful Concepts of BASIC 28
5 How to Talk with Your BBC Microcomputer 37
6 Read Data and Restore 44
7 Some More Hints on How to Write and Enter a Program 48
8 To Carry Out the Same Instructions 52
9 Jumping and Branching Statements 57
10 Conditional Branching 63
11 Counting and Splitting the Characters 67
12 Built-in Library Functions for Mathematical Calculations 70
13 Dimensions and Defined Functions 73
14 GET$ and INKEY$ Statements 85
15 Advanced Graphics 88
16 Making a Character 96
17 Sound 102
18 Teletext Control 108
19 Disc Handling 116
20 Sample Programs 120
21 Summary of Commands of the BBC Microcomputer 142
22 Computer Terminology 145

CORRIGENDA

Page & Line	*For*	*Read*
Page 14, l. 4	CURSOR CONTROL	CAP LOCK
" 64, l. 12	80 LET X= (-B+SQR(D))/2*A	80 LET X1 = (-B+SQR(D))/2*A
" 64, l. 13	90 LET X2 (-B-SQR(D))/2*A	90 LET X2 = (-B-SQR(D))/2*A
" 71, l. 20	50 G= DEG(2PI)	50 G= DEG(2*PI)
" 84, Fig.	DIM (4,3)	DIM (5,3)
" 88, l. 4 (from bottom)	MODE 0, 3, 4 & 5	MODE 0, 3, 4 & 6
" 89, l. 1	MODE 1 & 6	MODE 1 & 5
" 136, Program 23, l. 7	REACH"; CHR$(130) "VARIED HERITAGE";	RICH AND"; CHR$(130) "VARIED HERITAGE";
" 136, Program23, l. 5	AND SISTERS ";	AND SISTERS."; CHR$(132) "I LOVE MY COUNTRY";

CHAPTER 1

Electronic Computers and Their History

Computers have begun to make their presence felt, both directly and indirectly, in a growing number of spheres of our daily life. Our electricity bills, bank accounts, controlling devices at home and in industry, reservations in airlines, teaching, medicine, defence, printing, pay and accounts, business and scientific research – in short, almost every aspect of our lives, are affected by computers. The computer revolution is upon us!

As computers are making rapid inroads into our daily lives, only a very small minority of people with technical education in electronics really understand what these machines are made for and how they work.

In school education computers play a very important role. Teachers can use computers to teach their subjects more effectively. Some CBL (Computer-Based Learning) packages on graphs, chemical elements, transverse waves, longitudinal waves, molecular theory, etc. are already in use in our schools under the CLASS PROJECT. When the learning/teaching process is guided by a computer, it is known as CBL (Computer Based Learning) or CAI (Computer Assisted Instruction). The computer is also used in evaluation work and in the tabulation of marks.

WHAT IS A COMPUTER?

'A computer is an electronic device that has the ability to accept data, internally store and execute a programme of instructions, perform mathematical and logical operations on the data and report the results.'

A Brief History

The first mechanical calculator was developed by the Chinese as long ago as 2200 B.C. This was the abacus which has a rectangular wooden frame fitted with beads. It is still in use

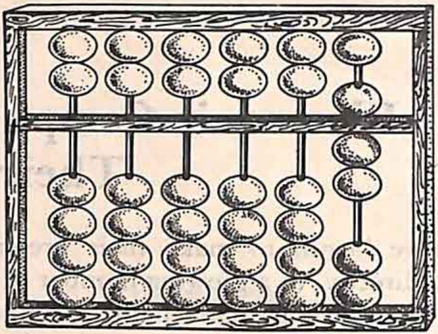


Fig. 1.1

in Japan, China, India and other countries for primary education. You might have seen one when you were in kindergarten.

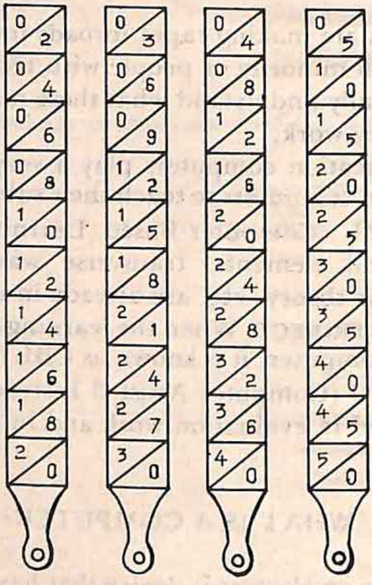


Fig. 1.2

John Napier, a distinguished Scottish mathematician prepared rods for multiplication around 1617. He called these rods 'bones'. He also developed logarithmic tables which were based on the laws of indices and were used for multiplication and division. Henry Briggs, a student of John Napier, prepared a common logarithmic table with base 10. The common logarithmic table is still in use and may be found in your Physics and Mathematics books.

Necessity is the mother of invention. A calculating machine was prepared by Blaise Pascal, a French mathematician, in 1641 when he was only eighteen. His father was an accountant and had a lot of calculation work to do, so Blaise Pascal prepared an ARITHMETIC MACHINE to help his father. This machine was fitted with gears and was very popular in those days. The Queen of Sweden was so fascinated by it that she placed an order for one such machine.

Fig. 1.3

In 1853 William Oughtred and Robert Bissakar simultaneously prepared a slide rule. This was very useful and remained in use upto the early 60s of this century. The slide rule facilitated most engineering calculations and it remained in use until electronic calculators made their appearance.

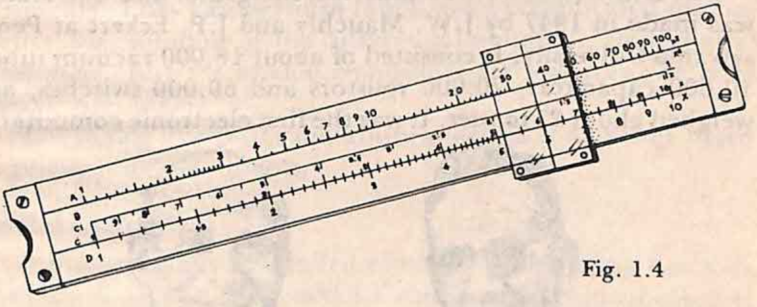


Fig. 1.4

Charles Babbage, an Englishman, was the first person to propose the basic concepts of the modern computer. He designed a Difference Engine in 1822. He was keen to incorporate the capability of storing intermediate results and recalling them when they were required in subsequent calculations. This is what the present computers are capable of. He was, however, unsuccessful in achieving this during his lifetime.

Fig. 1.5

Dr Herman Hollerith of the USA contributed a lot to the development of the computer. He was working in the census department. He invented a machine in which he used punched cards to store and tabulate census data/information. These cards were very popular and are even today a common medium for feeding information into the computer.

The world's first digital computer called MARK 1 was prepared jointly by Messrs International Business Machines and Howard Aikens and Grace Hopper of Harvard University in 1944. It had a limited memory and dissipated a very large amount of heat. It was also very slow.

The ENIAC (Electronic Numerical Integrator and Calculator) was made in 1947 by J.W. Mauchly and J.P. Eckert at Pennsylvania University. It consisted of about 18,000 vacuum tubes, 10,000 capacitors, 70,000 resistors and 60,000 switches, and weighed about 27 tonnes. It was the first electronic computer in

Fig. 1.6

the history of the computer. UNIVAC I was the first commercially available computer used for programming.

After the vacuum tube computer, transistorized computers were brought out in the late 1950s, followed by integrated-circuit computers in the mid-1960s. Then the microprocessor was developed. This is a single electronic chip and consists of a very large number of logical circuits. The use of the microprocessor reduced the size and price of the computer and enhanced its speed as well as capacity.

TYPES OF COMPUTERS

(i) Analog Computer
(ii) Digital Computer
(iii) Hybrid Computer

Analog Computer

In this computer numerical information (data) is presented in the form of a quantity, usually a voltage, that can vary continuously. This varying quantity is an analogue of the actual data, i.e. it varies in the same way as the data of the program.

Analog computers are used in the automation control of certain industrial operations and in various other scientific experiments. They can perform much more complicated mathematics, viz. solution of the differential equation, than digital computers. But the analog computers are less accurate than digital computers.

The Analog computer determines the desired answer based on physical measurements.

Digital Computer

An automatically controlled electronic calculating machine in which data is represented by the combination of discrete electrical pulses denoted by digit 0 and 1. Arithmetical and logical operations are performed on the data in accordance with the instructions supplied in the program. The instructions coded in binary form are analysed and processed by the central processing unit of the computer and the result is communicated to the operator. In simple words, the digital computer produces the desired result based on computation of numbers.

More than 90% of the computers worldwide are digital computers.

Hybrid Computer

A computer system that is equipped with both digital and analog capability. These computers are designed for certain specific functions and have various uses in the technical and scientific fields.

WHAT CAN A COMPUTER DO?

(1) Computers can perform Arithmetic Operations
(2) Computers can perform certain Logical Operations
(3) Computers can Store and Recall information
(4) Computers can Communicate with their operators
(5) Computers can direct themselves in a predetermined manner
(6) Computers can process large volumes of data
(7) Computers can interpret messages from remote locations

CLASSIFICATION OF COMPUTERS

According to use and size:

(1) Personal Computer
(2) Micro Computer
(3) Mini Computer
(4) Large Computer
(5) General Purpose Computer
(6) Special Purpose Computer
(7) Super Computer

Computer System

A computer system consists of the following components, all of which are necessary and of equal importance.

(i) **Hardware**: CPU, Printer, Disc drive, Tape recorder, Monitor/Console, Magnetic Floppy Disc, Program, Magnetic tape, etc.
(ii) **Software**: Programs

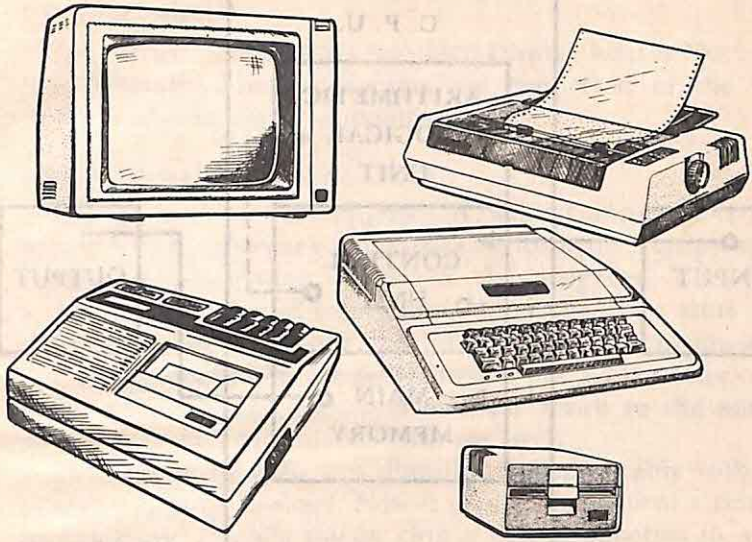


Fig. 1.7

HOW DOES A COMPUTER OPERATE?

The input and output units are the components with which the computer communicates with the user. The basic input signals can be set in two positions, like OFF-ON switches. The binary digits are 0 for OFF and 1 for ON positions. The input signals are first converted into binary signals and then processed by the computer, then reconverted into an understandable form. The memory unit is obviously for storage of data and programs. The arithmetic and logical units carry out the operations on the data that their names suggest. The control unit acts as a co-ordinator in the execution of a program. The arrows in the diagram represent the communication links between the different units. The central processing unit has three sub-units, namely, (i) control unit, (ii) main memory, (iii) arithmetic logical unit (ALU).

The diagram below illustrates the simple working and communication links between the different units of a computer system.

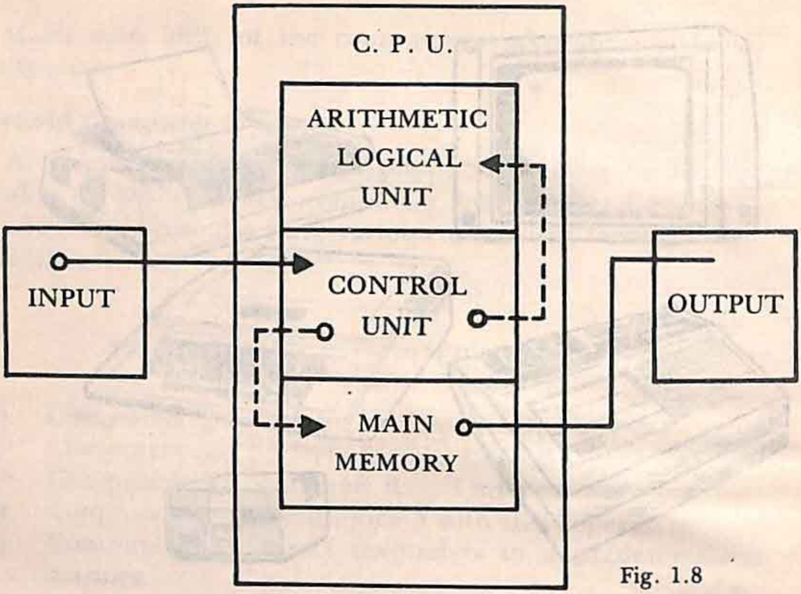


Fig. 1.8

WHAT DO YOU KNOW ABOUT I/O DEVICES?

Input Media

These are used to feed the data/information into the computer.

INPUT MEDIA	:	INPUT DEVICE
Magnetic Tape	:	Tape Drive
Magnetic Floppy	:	Disc Drive

Output Media

On these media, computers give out the result of the processed input data.

OUTPUT MEDIA	:	OUTPUT DEVICE
Magnetic Disc	:	Disc Drive
Magnetic Tape	:	Tape Drive
VDU Screen	:	VDU/Monitor
Paper (Stationery)	:	Printer

VDU (Monitor)

The Visual Display Unit or Video Display Unit is like a TV set. Generally home computers are connected to the TV, which is also known as a monitor.

CPU (Central Processing Unit)

It is the brain of the computer. A highly complex electronic device which supervises all activities within the computer, it reads the information (data) of the program. The CPU transfers each item of information from the main store and electronic circuits analyse the information. All arithmetical and logical operations are performed by the ALU. After complete processing the CPU transfers the result to the output device so as to communicate with the user.

The size of the CPU was diminished considerably with advancement in technology. Now it is possible to form a central processor on a single silicon chip a few millimetres in area. This tiny mastermind device is called a microprocessor.

The CPU consists of three parts: (i) Main memory; (ii) Control unit; (iii) Arithmetic & Logic unit (ALU).

Small and Large Computers

A computer is classified as small or large on the basis of its storage capacity. A computer with a very small memory capacity (eg. 16K or below) is called a small computer whereas one with large memory capacity (256K, etc) is called a large computer. Memory is measured in Byte/Kilobyte (K). 1K = 2 ^ 10 Byte, i.e. 1K = 1024 bytes. The BBC microcomputer has got a 32 K memory.

Bit & Byte

The CPU is made up of numerous tiny electronic circuits (called gates) which distinguish between the two levels of electronic current flowing through them, namely zero and one. All data, however complex, is represented by a combination of the two digits zero and one (Binary digits). The fundamental units (zero & one) in a binary system are called 'bits'; and a collection of multiple of eight bits is called a 'byte'. The abbreviation of a BInary digiT is BIT. Normally, a single character will occupy 1 byte memory.

High Level Computer Languages

There are many different high level languages that are used for computer programming. The most basic of these is the machine language which is not a high level language. Some common high level languages are listed below.

NAME OF LANGUAGE	EXPANSION	PURPOSE
BASIC	Beginners All-purpose Symbolic Instructional Code.	General
FORTRAN	FORmula TRANslation	Engineering & Scientific
COBOL	COmmon Business Oriented Language	Business
CAL	Conversational Algebraic Languages	Scientific
PL/I	Programming Language I	Business & Scientific
PASCAL	Named after French scientist BLAISE PASCAL	Scientific & General

Limitations of the Computer

The computer cannot exercise judgement or commonsense, and it must be clearly instructed in the program as to how to handle every contingency. What to do if the answer of an intermediate calculation becomes zero, or when particular exceptions to the anticipated results occur. Each eventuality must be analysed completely or the program will not work satisfactorily (although this may not be immediately apparent).

EXERCISES

1. What is an electronic computer?
2. Distinguish between:
   (a) Analog and digital computers;
   (b) bit and byte;
   (c) input and output devices;
   (d) CPU and VDU.
3. Who invented the slide rule?

4. Write down the full form of
   - (a) CPU
   - (b) COBOL
   - (c) BASIC
   - (d) FORTRAN
   - (e) ALU
   - (f) BIT
   - (g) ENIAC
   - (h) PL/I
   - (i) I/O device
   - (j) VDU
5. What was the name of the first electronic computer?
6. How many bytes are there in a kilobyte of memory?
7. What does a computer do?
8. What are the different types of computers?
9. What do you understand by the term small and large computers?
10. What is a microprocessor?

CHAPTER 2

The Keys of the BBC Micro BASIC Computer Language

BASIC is an acronym for Beginners All Purpose Symbolic Instructional code. This was first invented by Prof. J.G. Kemeney and T.E. Kurtz of Dartmouth college, New Hampshire, USA in the early 1960s and was implemented in 1964. Since then many BASIC versions have been developed. One of them is 'standard BASIC' used by BBC Micro-computers in Indian schools.

Most of the instructional statements of BASIC are similar to those of algebra and a simple version of the English language. It is, as a result, easy to learn and understand.

Merits of BASIC

1. Easy to learn;
2. suitable for conversational programming;
3. simplified grammar;
4. suitable for carrying out arithmetic and logic operations;
5. universally available;
6. used for both Mathematical and Business purposes;
7. easy to edit;
8. the result may be represented in sophisticated form;
9. provision for filing of program and data;
10. primary language of Microcomputer 'faddist'.

Demerits of BASIC

1. Lengthy programs;
2. slow in execution;
3. cannot be used for very advanced engineering and scientific programs.

KEYS OF THE BBC MICROCOMPUTER

There are 73 keys and a space bar on the keyboard of the BBC Microcomputer. Fortyeight of them are used for typing the characters but the other twentyfive keys are used for special pur-

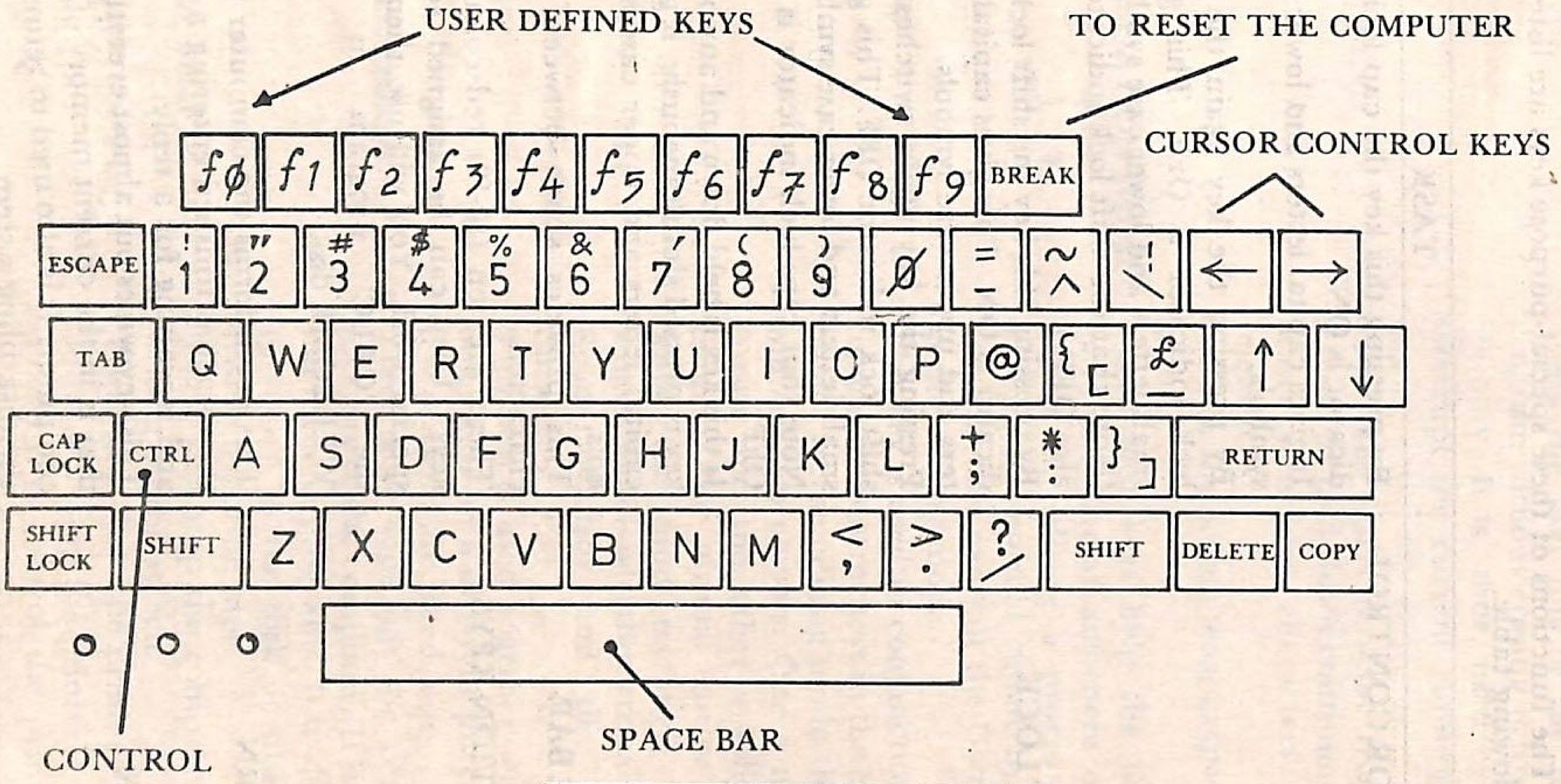


Fig. 2.1 KEYBOARD OF BBC COMPUTER

poses. The functions of these special-purpose keys are listed in the following table.

KEYS	TASK
CURSOR CONTROL	By pressing this key the cap lock indicator is ON. It gives capital letters and lower case symbols. By pressing the key again the cap lock indicator is OFF. This gives small letters and lower case symbols. (Note that the shift lock indicator is also OFF.)
SHIFT LOCK	By pressing this key the shift lock indicator is ON. This gives capital letters and upper case symbols. Pressing the key again switches the shift lock indicator OFF. This gives small letters and lower case symbols. Note that caps lock indicator is also OFF.
SHIFT	If this key is held down and any other key is pressed simultaneously, it gives capital letters and upper case symbols.
SPACE BAR	This provides space between two characters.
FUNCTION KEYS	These are ten (f0-f9) red-coloured keys. They can be assigned some special work. You will use them in the 'CALC' program in the WELCOME disk.
RETURN	This key informs the computer that the user has finished entering a line and is waiting for a reply.
BREAK	This key wipes out almost everything that is in the current memory of the computer. It is also used to return to the disk-filing system.

ESCAPE : Takes the user out of the present program without erasing it from the memory. It is, thus, used to stop/interrupt the current running program.

DELETE : Erases the last character from the screen. If held down continuously it erases further characters until released.

TAB ; Used in the WORD PROCESSING program.

CURSOR CONTROL → ← ↑ ↓ : These four keys help the user to move the Cursor anywhere on the screen for editing a program. By moving the normal cursor by these keys the user can put the Computer in the editing mode.

COPY : This key is used in conjunction with the cursor control keys in the editing mode. In the editing mode the user will see two cursors. One a normal cursor and the other a solid rectangle. The normal cursor reads while the solid rectangle cursor writes. That is why they are called the Reader Cursor and the Writer Cursor.

CTRL : This key is used together with the other keys for a different purpose. A few of them are listed here.

CTRL BREAK : These keys wipe out the complete memory of the computer. It is essential for long programs. It is also used to return to the disk or disk-filing system from the tape.

CTRL SHIFT : These keys stop the computer from writing on the screen.

CTRL B : These keys start the printer.

CTRL C : These keys stop the printer.

CTRL N : These keys control the printing of line numbers on the screen. It is essential to see a lengthy program page by page. By using SHIFT the user can see the next page. Thus we can say that this puts the computer in paged mode.

CTRL O : These keys enable return to original mode from the paged mode.

CTRL G : These keys make the internal speaker produce a short noise.

CTRL ^ : These keys move the cursor to the top of the text area.

CTRL I : These keys move the cursor by one character position.

CTRL U : These keys move the cursor down one line.

CTRL P : These keys clean the graphic area of the screen to the graphic background colour.

SHIFT f0 to F9 : These keys are used to get a colour effect in MODE7.(See Users Guide for details.)

CHAPTER 3

Flow Chart and Algorithm (A First Step in Programming)

A computer cannot be used to solve a problem until a method or solution has been evolved and a detailed procedure has been prepared by the programmer. It is essential, therefore, that the programmer has a clear cut concept of the problem-solving procedure and sufficient deductive and reasoning skills.

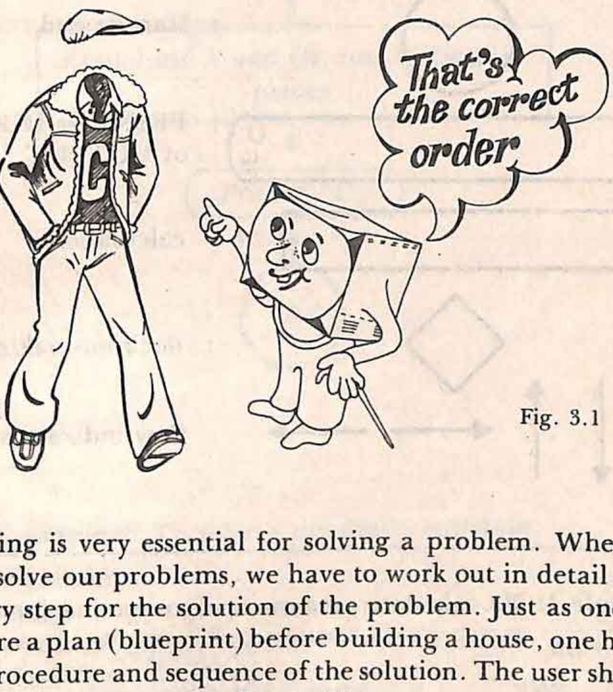


Fig. 3.1

Planning is very essential for solving a problem. When we want to solve our problems, we have to work out in detail each and every step for the solution of the problem. Just as one has to prepare a plan (blueprint) before building a house, one has to list the procedure and sequence of the solution. The user should prepare a strategy to solve the problem.

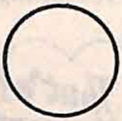
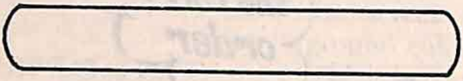
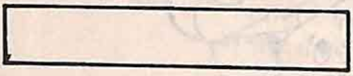
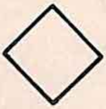
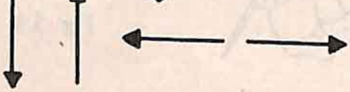
The sequence of the instructional steps for solving a problem is called algorithm.

A flow chart is a diagrammatical representation that illustrates the sequence of operations to be performed to arrive at the solution. The instructions are placed in boxes and are connected by arrows to indicate the order of execution.

Merits of a Flow Chart

1. A flow chart is an important instrument in the development of the algorithm.
2. It facilitates the preparation of a computer program.
3. It can be used to translate an algorithm in any programming language.

The commonly used box symbols and their meanings are given below:

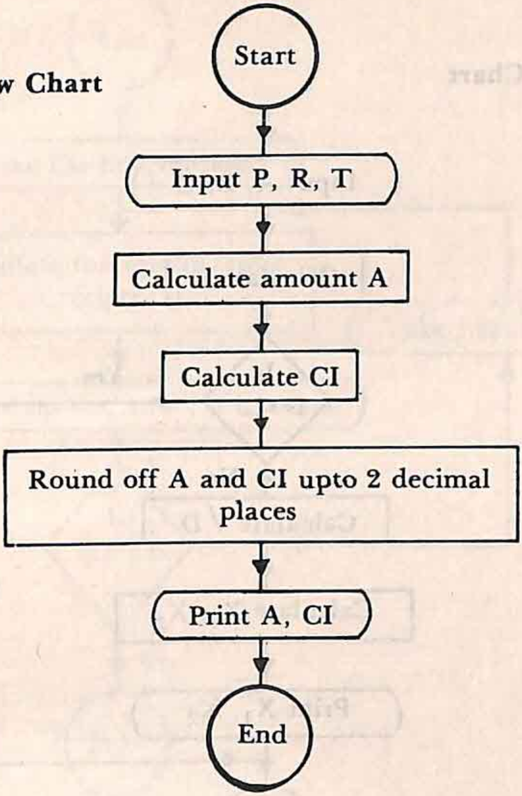
SYMBOLS	: MEANING
	: start or end
	: PRINT or INPUT or OUTPUT
	: calculation
	: decision-making
	: flow indicators

Example 1: To calculate amount and compound interest.

Algorithm

```
1   Start
2   Read P,R,T
3   Calculate $A = P(1 + R/100)^n$
4   Calculate $CI = A - P$
5   Round off upto two decimal places
6   Print A, CI
7   Stop
```

Flow Chart

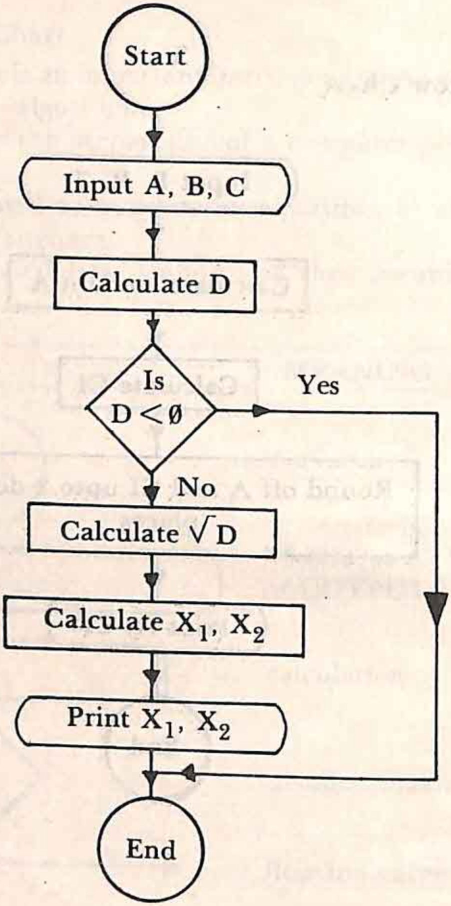


Example 2: To solve a quadratic equation.

Algorithm

1 Start
2 Read coefficient of X^2, coefficient X and the constant term; A, B, C
3 Calculate $D = B^2 - 4AC$
4 If $D < 0$ then print no real roots and stop
5 If $D > 0$ then calculate X1 and X2,
$\left[\text{using } X = \dfrac{-B \pm \sqrt{D}}{2A}\right]$
6 Print out the results
7 Stop.

Flow Chart

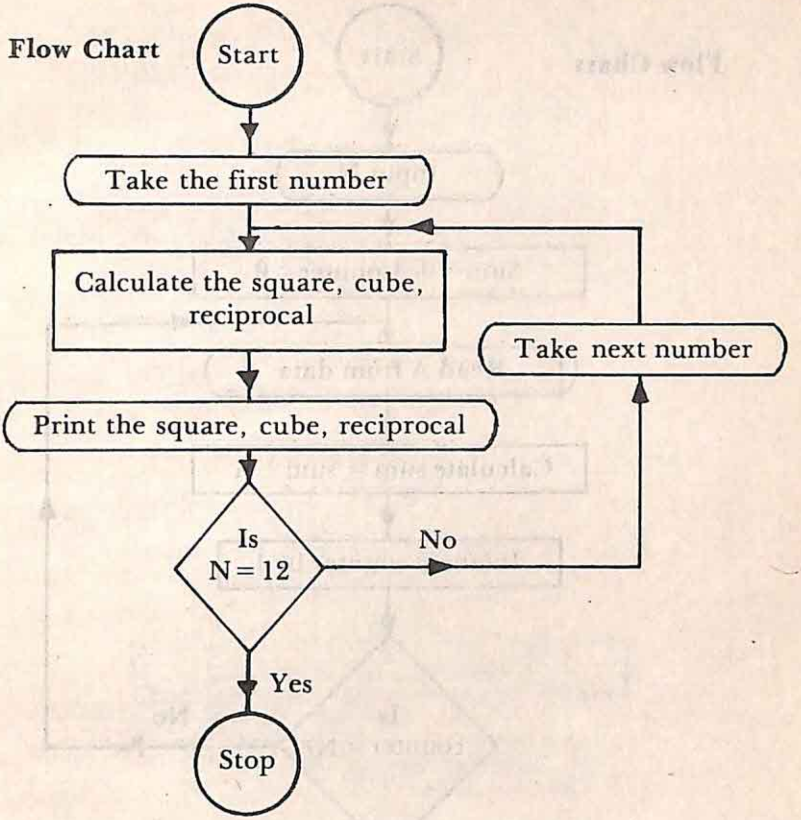


Example 3: To prepare a table of squares, cubes and reciprocals of numbers from 1 to 12

Algorithm

1 Start
2 Read a number, say N
3 Calculate N^2, N^3 and 1/N
4 Print
5 If N < 12 then repeat steps 2, 3 and 4
6 Stop.

Flow Chart


Start
Take the first number
Calculate the square, cube, reciprocal
Print the square, cube, reciprocal
Is N = 12
No
Take next number
Yes
Stop

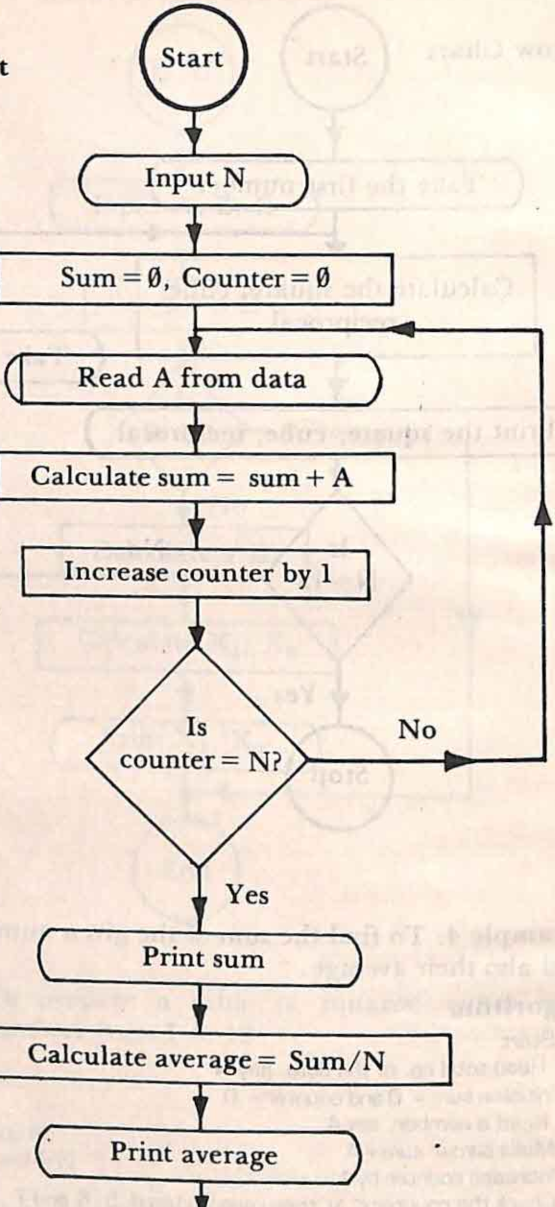
Example 4: To find the sum of the given numbers in the data and also their average.

Algorithm

```
1 Start
2 Read total no. of the data, say N
3 Initialise sum = 0 and counter = 0
4 Read a number, say A
5 Make sum = sum + A
6 Increase counter by 1
7 Check the counter < N, then repeat step 4, 5, 6 and 7
8 Print out sum
9 Calculate average
10 Stop.
```

15350

Flow Chart

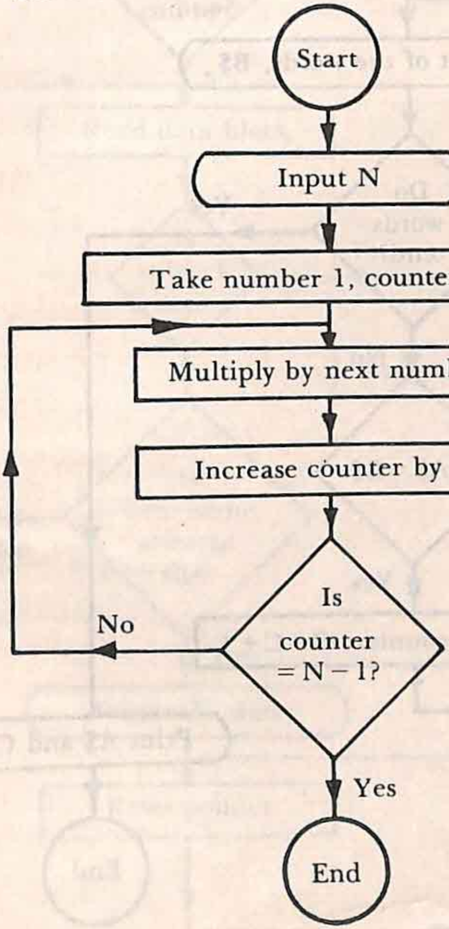

Start
Input N
Sum = 0, Counter = 0
Read A from data
Calculate sum = sum + A
Increase counter by 1
Is counter = N?
No
Yes
Print sum
Calculate average = Sum/N
Print average
End

Example 5 To calculate factorial of N (i.e. $\underline{|N}$)

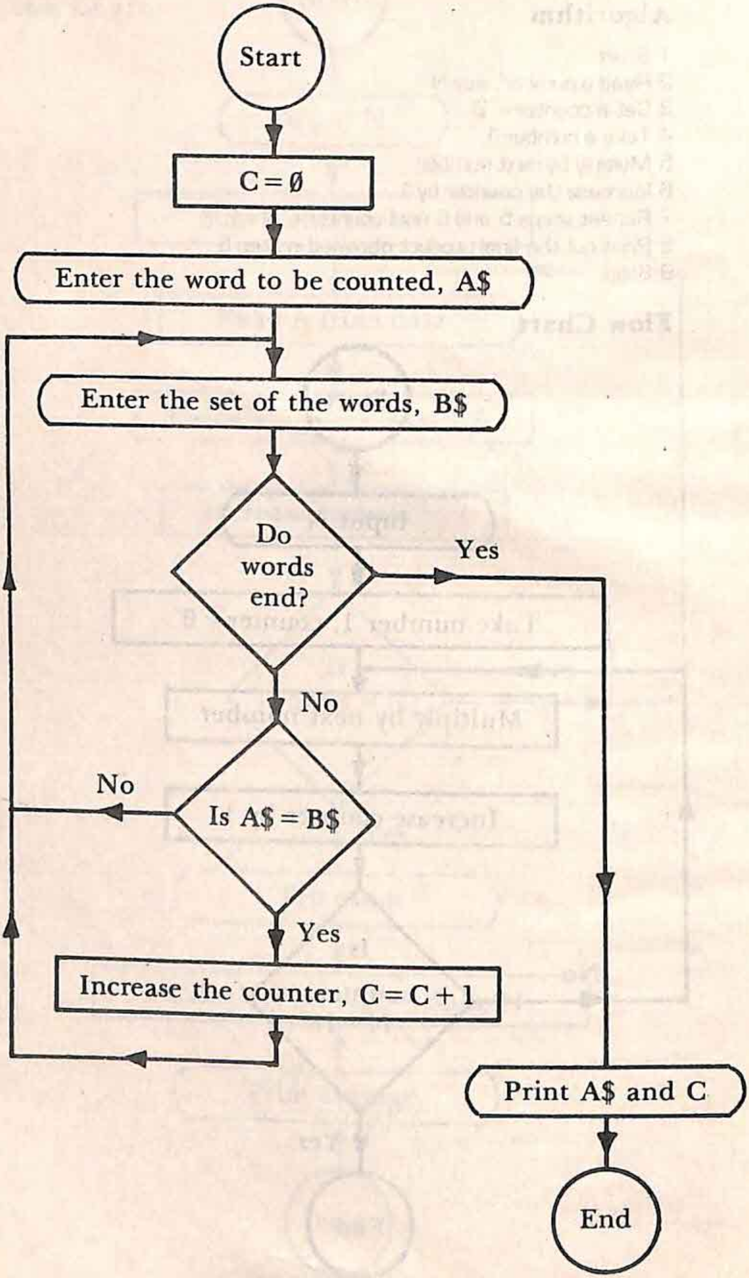
Algorithm

1 Start
2 Read a number, say N
3 Set a counter = 0
4 Take a number 1
5 Multiply by next number
6 Increase the counter by 1
7 Repeat steps 5 and 6 until counter = N – 1
8 Print out the final product obtained in step 5
9 Stop.

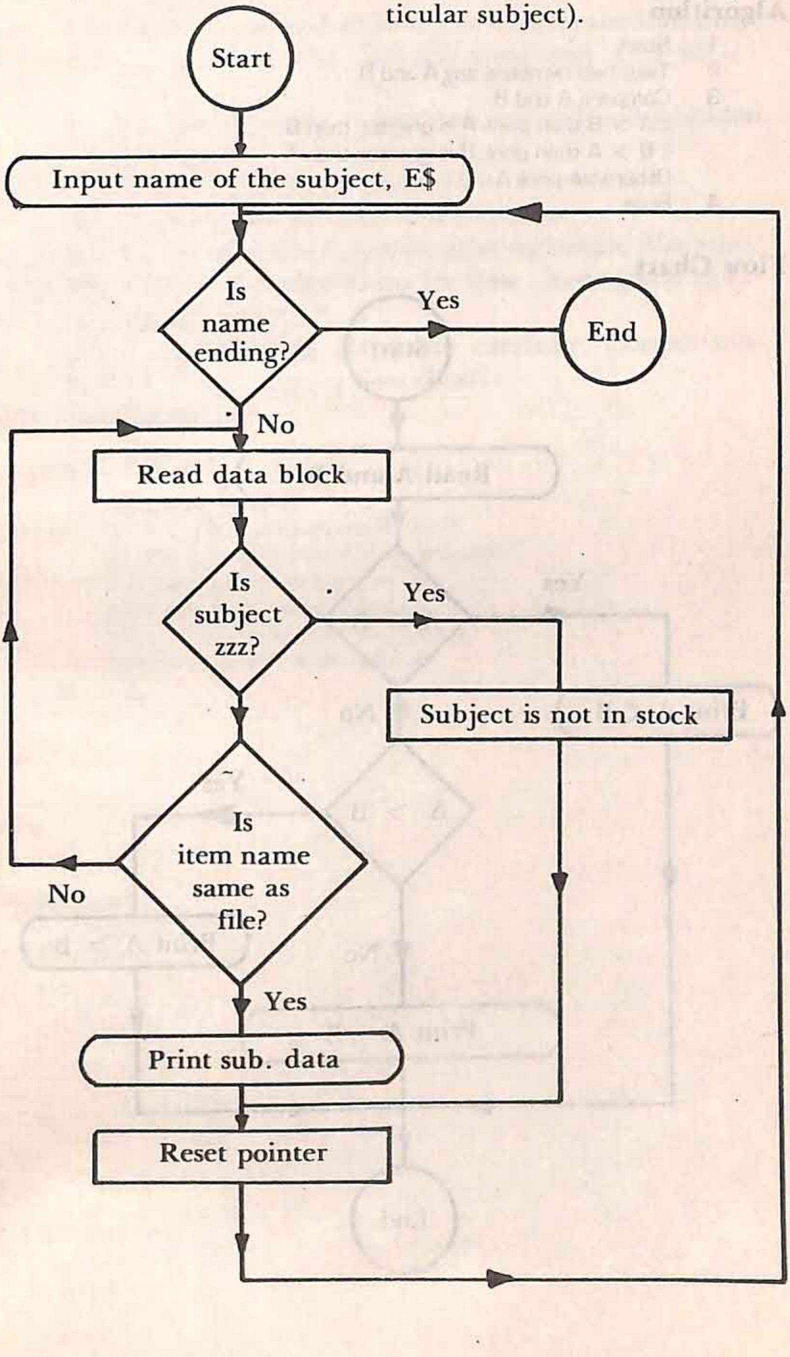
Flow Chart



Example 6 To count the desired word in the set of words.


Start
C = 0
Enter the word to be counted, A$
Enter the set of the words, B$
Do words end?
Yes
No
Is A$ = B$
No
Yes
Increase the counter, C = C + 1
Print A$ and C
End

Example 7 Flow chart for stock query (say, books on a particular subject).

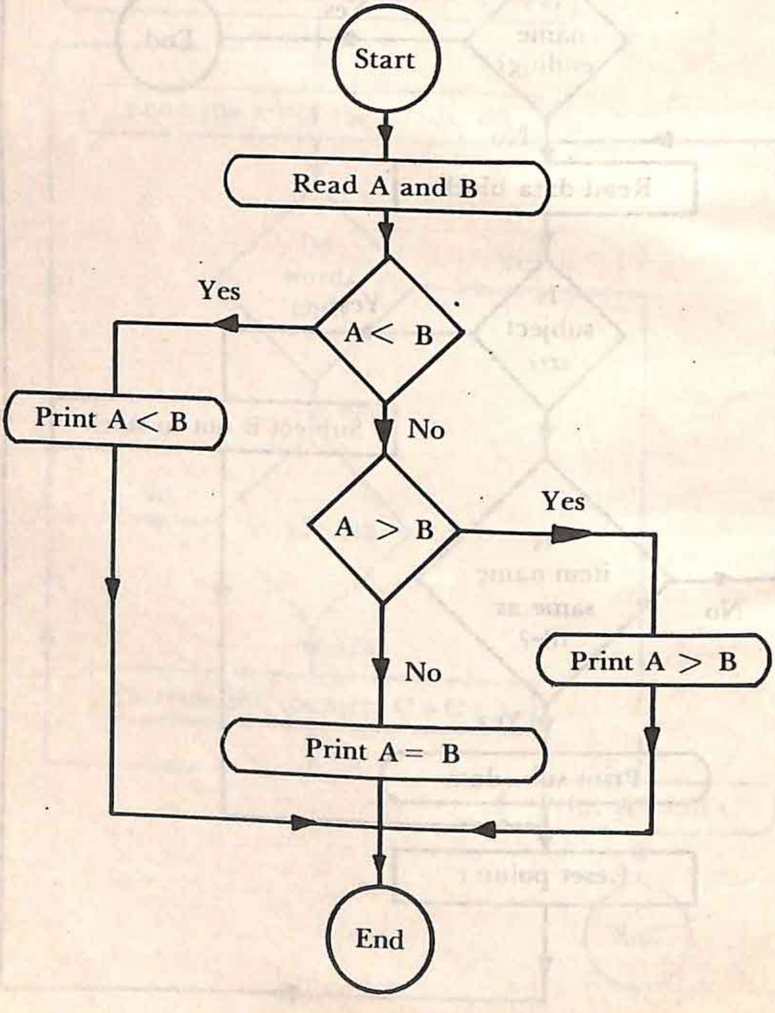


Example 8 To find the order of two numbers.

Algorithm

```
1    Start
2    Take two numbers say, A and B
3    Compare A and B
     if A > B then print A is greater than B
     if B > A then print B is greater than A
     Otherwise print A = B
4    Stop.
```

Flow Chart



EXERCISES

1(a) Draw flow charts and write algorithms for the following:
   1. To calculate the circumference and area of a circle.
   2. To calculate the compound interest compounded half yearly.
   3. To calculate factorials of 10.
   4. To calculate the area of a rhombus.
   5. To calculate the net price after successive discounts.

(b) Write down the algorithms for flow charts given in examples 6,7 and 9.

2 Read the following algorithm carefully. Correct mistake(s), if any. Draw a flow chart.

Algorithm

```
1   Start
2   Input three numbers, say A, B, C.
3   Compare A with B
     [i]  If A > B, then compare A with B.
     [ii] If A > C, then print A is the greatest.
    Otherwise print C is the greatest.
    If 3[i] fails, then compare B and C.
    If B > C, then print B is the greatest,
    Otherwise print C is the greatest.
4   Stop.
```

CHAPTER 4

Some Useful Concepts of BASIC

Characters on the Keyboard

The BBC Microcomputer has a set of characters which consists of letters,. digits, punctuation marks and other symbols. They are classified in three subsets.

1. Letter of the alphabet
   A,B,C,......,Z ; a,b,c,......z
2. Digits
   0,1,2,3.........,9
3. Special characters (symbols)

+	plus (operator for addition)
–	minus (operator for substraction)
*	asterisk (operator for multiplication)
/	slash (operator for division)
^	upward arrow (operator for exponentiation)
(	opening bracket
)	closing bracket
,	comma
;	semicolon
:	colon
$	dollar
%	percentage
#	hash
.	point
<	is less than
>	is greater than
=	is equal to
—	dash
”	double quotation marks
£	pound
&	and
!	exclamation mark
@	at the rate of
?	question mark
’	apostrophe
[	opening bracket

]	closing bracket
(	opening parenthesis
)	closing parenthesis
>=	is greater than or equal to
<=	is less than or equal to

Constants

Constants are numerals and words which do not change their values during the program. There are two types of constants.

1. **Numeric Constants** Numeric quantities are those which refer to numbers.

Example 67.7, 4657.478, 23E12, 0.21345, −2145.45

The magnitude of numbers that can be processed by the BBC Microcomputer ranges from 10^38 to −10^38.

BASIC allows a number to have as many as 8 or 9 significant figures. Commas are not permitted in numeric constants and the positive sign is optional.

2. **String Constants** A string constant is any set of valid BASIC characters enclosed in double quotation marks. A string constant may have a letter number or space with quotation marks.

Example “NARENDER”, “DEVENDRAMOHAN SINGH”, “BHOPAL”, “KENDRIYA VIDYALAYA SANGATHAN”, “DECEMBER 25, 1982”, “4567”, “P + Q/R”

RELATIONAL SYMBOLS AND THEIR USAGE

Relational Symbol	Usage	Meaning
=	P=Q	P is equal to Q
<	P< Q	P is less than Q
>	P > Q	P is greater than Q
< = or =<	P < =Q	P is less than or equal to Q
> = or = >	P=>Q	P is greater than or equal to Q
>< or < >	P< > Q	P is not equal to Q

Note The symbols = (is equal to) and < > (is not equal to) are also used for string variables. Such as

```
10 LET A$= "CALCUTTA"
```

```
10 LET K$ < > “RAM”
```

Variables

BASIC deals with three types of variables.

1. **Numeric variables** A numeric variable is generally represented by a single letter followed by a single digit but the BBC Micro allows one to use more letters such as RC, NUMBER, MSS, K9, etc.

2. **String variables** In the BBC Micro the string variables are defined by a set of alphabets followed by a dollar symbol ($). For example NAME$, A$, RAM$, LESSON$, etc. But you cannot use the string variable LETTER$ since LET is a BASIC instruction. So avoid such strings which can confuse the computer.

3. **Integer variables** Integer variables are represented by a single letter or single letter followed by a single digit terminated by a percentage symbol (%), such as A%, S9%, K3%, etc.

MATHEMATICAL OPERATORS

Symbol	Operation
+	Addition
−	Subtraction
*	Multiplication
/	Division
^	Exponentiation

Expressions

An expression is a symbol or group of symbols connected by operators. These symbols may be numbers, strings, variables and functions.

Here it is to be noted that two arithmetic operators cannot be used consecutively in the same expression, such as A + − B, X ^ *Y, but a unary minus is valid, as in P* − Q.

Order of Execution

BASIC assigns the highest precedence to exponentiation. The next precedence is given to unary minus and then multiplication and division (both of which have the same order). The lowest order is given to addition and subtraction (both of which have the same order). But the expressions in closed brackets are evaluated first.

Statement Numbers

Statement numbers are also known as the line numbers of the instructions. These line numbers are taken from 1 to 9999. Generally, statement numbers are taken in multiples of 10 which allow insertion of some missing line numbers. The computer executes the program line by line, according to these line numbers.

SOME SYSTEM COMMANDS AND BASIC STATEMENTS WITH THEIR USES

KEYWORDS	TASK
*CAT	To get a catalogue of the programs in the disc
*TAPE	To connect the tape drive to the computer
*DISK or *DISC	To connect the disc drive
*DELETE	To remove a program from the memory of the disc or tape
*KEY	To program one of the user-defined keys
*OPT	Determines how the computer reacts to errors during loading of cassettes and the amount of detail given during cassette operations
AUTO	To automatically get program number lines as a program is written. The first number and spacing between the line numbers can be specified
LOAD	To transfer from the cassette or disc to the computer memory
RUN	To execute the program
CHAIN	Equivalent to LOAD+RUN
LIST	To display the list of the program instructions which is currently in the memory
NEW	To remove the program from the computer's memory
OLD	To recover a program deleted from the

	memory by typing NEW or pressing BREAK
CLS	To clear the text area of the screen
CLG	To clear the graphic area of the screen
RENUMBER	To convert all line numbers into multiples of 10 along with the GOTO numbers in the program
SAVE	To record the current program on the disc or cassette

Fig. 4.1

BASIC PROGRAMMING

The functioning of the computer is controlled by a set of orderly instructions. A complete set of orderly instructions is called a COMPUTER PROGRAM. The person who is preparing the program is known as the PROGRAMMER and the method/ process of preparing a program is known as PROGRAMMING. The language which is used in communicating computer instruction is known as the PROGRAMMING LANGUAGE. The program in BASIC language is called a BASIC PROGRAM.

How to Start Your BBC Microcomputer

Plug in the sockets and switch on the electricity. Now switch on the computer and the VDU, and then the BBC Micro. You will hear a two tone sound of greeting 'HEL-LO'.

Now you will see the following message displayed on the VDU screen

```
BBC computer 32K
Acron DFS
BASIC
>–
```

This message tells us the name of the computer "BBC computer" which has 32K (kilobytes) memory. Acron is the name of the manufacturing company. DFS stands for disc filing system. This computer follows the BASIC language. The symbol > is called the 'prompt' which means that the computer is ready. The flashing line is called the 'cursor', and it shows you where on the screen your typed instructions will appear.

Now the computer is ready for your work and it is your turn to assign some work to it in its language, i.e. BASIC.

Entering a Program

Whenever you start your Microcomputer you will find the prompt symbol and the cursor on the VDU.

Then the user has to type the line number first, followed by the BASIC statement. The typed information will not go into the computer memory until it is terminated by pressing the RETURN key. As soon as you press the RETURN key the prompt (>) symbol and the cursor will appear at the beginning of the next line. Again, the user has to type the next line number followed by BASIC statement and terminate by pressing the RETURN key.

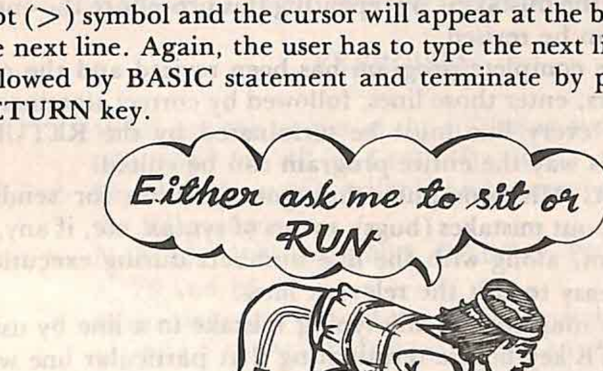

Either ask me to sit or RUN

Fig. 4.2

This procedure is continued until all the program statements have been entered and placed in the computer's memory.

The user will see the prompt symbol after pressing the RETURN key for the last time.

Now the user has to decide what to do with the programming, viz. RUN, LIST, or SAVE.

Editing a Program

It is quite possible that some mistakes may occur in the process of writing and entering a program. It would be very tiresome and time consuming to rewrite and re-enter a whole program. To correct mistakes the computer provides an editing facility. For this the user has to type LIST terminated by RETURN. The whole program appears on the VDU. If your program is lengthy then,by pressing the 'CTRL & N' keys together before typing LIST and terminating it with the RETURN key, you will control the line numbers. Now revise the program carefully and note down the mistakes. By pressing SHIFT, the user will get the next few lines of the program. Note down the mistakes. By repeating this procedure the entire program can be revised.

After the complete program has been revised and the prompt appears, enter those lines, followed by correct statements. (Note that every line must be terminated by the RETURN key.) In this way the entire program can be edited.

The BBC Microcomputer has got provision for sending messages about mistakes (bugs), errors of syntax, etc, if any, in the program, along with the line numbers during execution, making it easy to edit the relevant lines.

The user may correct any typing mistake in a line by using the DELETE key before terminating that particular line with the RETURN key.

To delete any line of the program, type the line number only and press the RETURN key.

POINTS TO REMEMBER FOR EFFICIENT PROGRAMMING

1. Master the rules of BASIC thoroughly.

2. Programming is like mathematics—you must be logical, have a plan (flow chart) and attempt to carry it out.
3. Learn looping and branching thoroughly.
4. Use REM statements at all important points in your program, so that it is easy to detect errors and revise them at a later date.
5. Be a perfectionist.
6. Be careful about your use of syntax and extra space. Very small mistakes are enough to crash (i.e. ruin) your program.
7. Allow yourself enough time on the computer.
8. Don't lose your temper when there is a bug in your program. Record its LIST on the printer and revise later when you are fresh.
9. Don't program when you are tired.
10. Don't try to read the entire computer manual at one time.
11. Don't confuse the digit zero (0) and alphabet O.
12. Care should be exercised in handling IF...THEN, FOR..TO..NEXT and GOTO statements.

EXERCISES

1. What are the names of the following symbols?
   (a) / (b) * (c) # (d) $ (e) ' (f) " (g) !
2. Separate the string variable and numeric variable from the following:
   NAME$, N, NAME, N9$, RADIUS, P2, CODE$, PQ, S, Y$ and E$
3. Write down the uses of the following system commands (a) *CAT (b) *DELETE (c) *DISC (d) *TAPE
4. Define each of the following terms: (a) PROGRAMMING LANGUAGE (b) PROGRAM (c) PROGRAMMER
5. Write down the uses of the following statements: (a) RUN (b) LIST (c) LOAD (d) CLS (e) CHAIN
6. Point out the error in the following mathematical expressions: (a) A + /B (b) P + 7Q (c) (X*Y)/0.0 (d) (A − B*^2) (e) (X*(Y − Z)) (f) M + A$ − 7

(g) (P + Q)*(− 3)^0.5

7. What are the smallest and largest line numbers in a program?
8. What are the following symbols called: (i) & (ii) > (iii)£ (iv) @?
9. How do you delete a line from a given program?
10. Write the names of the following variables: (i) A$ (ii) N (iii) A%.

CHAPTER 5

How to Talk with Your BBC Microcomputer?

There are a wide range of instructions in the BASIC language used with the BBC microcomputer. We will start with the PRINT and LET statements. Type the following:

>PRINT "HELLO! I AM MOHAN." (press RETURN key)

You will see the message

HELLO! I AM MOHAN.

Now you want to do some calculations which involve arithmetic operations. Simply giving instructions such as >7 + 9 will not get any response. Try it.

>7 + 9 (press RETURN key)

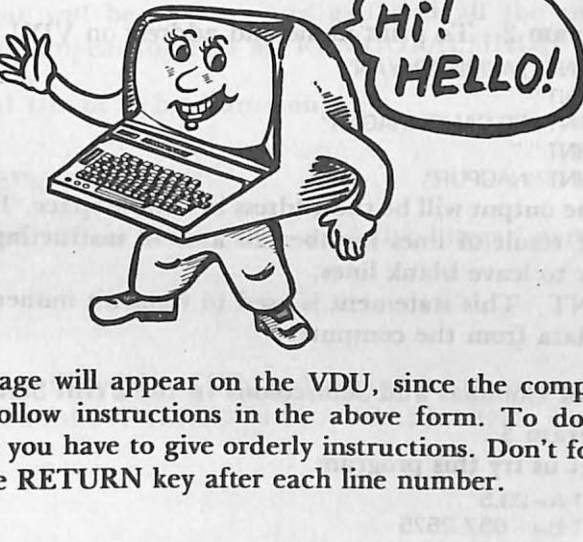


Fig. 5.1

No message will appear on the VDU, since the computer does not follow instructions in the above form. To do this calculation you have to give orderly instructions. Don't forget to press the RETURN key after each line number.

```
10 LET A=7
20 LET B=9
30 LET C=A+B
40 PRINT C
RUN
```

Now you will see the sum of 7 + 9 (which is obviously 16 !) displayed on the screen.

The above program can be written, as follows, and will also give the same output:

```
10 LET A=7
20 LET B=9
30 PRINT A+B
RUN
```

Here in line 30 PRINT A + B will find the sum of A and B and also print out the sum.

A FEW PROGRAMS FOR YOU TO TRY

Program 1 To calculate the area and circumference of a circle

```
10 LET R=10
20 LET A= (PI) *R^2
30 PRINT "AREA OF THE CIRCLE=";A;" sq units"
40 LET C=2*(PI) *R
50 PRINT "CIRCUMFERENCE OF THE CIRCLE=";C;" units"
```

To get the result of the program type RUN and press the RETURN key. Note: The value of PI = circumference/diameter (i.e. 3.1416....)

Program 2 To print name and address on VDU >

```
10 PRINT "LAXMINARAYAN"
20 PRINT " "
30 PRINT "30 GANDHINAGAR"
40 PRINT " "
50 PRINT "NAGPUR"
```

The output will be the address in double space. The spacing is the result of lines number 20 and 40 instructing the computer to leave blank lines.

PRINT This statement is used to transmit numerical or string data from the computer.

Use of Commas and Semicolons in the Print Statement

Program 3

Let us try this program

```
10 LET A=23.5
20 LET B=-657.2675
30 LET C=21.32
40 LET D=-5.23
50 LET E=0.5643808
60 LET F=45
70 LET G=435.218769
```

```
80 PRINT A,B,C,D,E,F,G
```

In the above program line numbers 10 to 70 assign the value to numeric variables A,B,C,D,E,F and G. We will discuss the LET statement later in detail. The statement 80 will give the output of values A and D in a line and the rest, E, F and G, in the next line, zonewise as follows:

```
      23.7      657.2675          21.32      -5.23
 0.5643808            45     435.218769
```

If the line number 80 is modified as follows:

```
80 PRINT; A;B;C;D;E;F;G
```

The output will be quite different. Try it yourself.

The output will be in a single line without a gap. But the use of semicolons is very essential in line 30 of PROGRAM 1 since we want only one space gap in between two words or numbers. Remove the semicolons in line 30 of the PROGRAM 1 and see the difference.

Program 4

```
20 PRINT "RAM";
30 PRINT "GOPAL";
40 PRINT "SINGH";
```

The output will be without any gap and all the print statements will appear together as: RAMGOPALSINGH

Some Special Use of Print Functions

Program 5

```
10 LET A%=3218.786
20 PRINT A%
```

The output will be 3218 (i.e. only the integer part of 3218.786).

Program 6

```
10 VDU 74
```

The output will be J (i.e. J followed, of course, by the prompt symbol >) since the AMERICAN STANDARD CODE (ASC) for J is 74.

Program 7

```
10 PRINT ASC ("R")
```

The output will be 82, since the ASC for 82 is R.

Program 8

```
10 PRINT CHR$ (82)
```

The output will be R, since the ASC for R is 82.

Program 9

```
10 LET A$="SWATI"
20 LET B$="SHARMA"
30 PRINT A$+ " " + B$
```

The output will be

```
SWATI SHARMA
```

Program 10

```
10 PRINT "TECHNICAL TEACHERS TRAINING INSTITUTE"
20 PRINT ' ' "BHOPAL"
30 PRINT ' ' ' "PIN-462002"
```

In line 20 there are two apostrophes, so BHOPAL will appear two lines after the first. Similarly PIN − 462002 will appear, leaving three lines after BHOPAL.

LET Statement

This statement is used to assign a numerical or string value to a variable.

Program 11

```
10 LET A=25
20 LET N$="MOHAN LAL"
30 LET A=A+5
40 LET A=A/3
50 LET PRINT N$; " has ";A;" mangoes. "
```

Explanation:

Line 10 assigns the numerical value of A = 25

Line 20 assigns the string value of N$ = MOHAN LAL

Line 30 increases the value of A by 5

Line 40 divides the value of A by 3

The output of 50 will appear on the VDU as

```
MOHAN LAL has 10 mangoes.
```

INPUT Statement

This statement instructs the computer to wait for some data from the keyboard which is then used in a program. It thus permits one to enter the data while the program is running.

It waits for values of the variables which are to be typed and are separated by commas. Pressing the RETURN key allows one to further continue the program.

Let us try these programs.

Program 12 To converse with the computer

```
10 PRINT "WHAT IS YOUR NAME";
```

```
20 INPUT NAME$
30 PRINT "HOW OLD ARE YOU (ENTER NUMBER OF YEARS)";
40 INPUT Y
50 PRINT " HELLO! "; NAME $;
60 PRINT " YOU ARE "; Y;" YEARS OLD "
```

It is notable here that we may combine 10 and 20, also 30 and 40 to obtain the same output:

```
10 INPUT " WHAT IS YOUR NAME "; NAME $
30 INPUT " HOW OLD ARE YOU (ENTER THE NUMBER OF YEARS)" ; Y
```

By keeping 50 and 60 as they are we get the same output.

Program 13 To calculate interest for a given principal, rate and time.

```
10 INPUT "ENTER PRINCIPAL IN RS "; P
20 INPUT " ENTER RATE OF INTEREST "; R
30 INPUT " ENTER TIME IN YEARS "; T
40 LET I=(P*R*T)/100
50 PRINT " INTEREST Rs= "; I
```

You may add a line to restart the same program.

```
60 PRINT "TYPE RUN TO RESTART"
```

Now remember that INPUT is the BASIC way of saying 'put in'.

The INPUT statement will reject leading spaces and everything after a comma unless the entry is in quotation marks.

```
100 PRINT "ENTER TWO NAMES SEPARATED BY A COMMA"
110 INPUT P$
120 PRINT P$
```

If you enter ANIL and SHARMA during RUN, the output of line 120 will be:

```
ANIL
```

If we use the INPUT LINE line statement we get everything that is typed, including spacing and commas. Try the following program.

```
100 PRINT "ENTER A SENTENCE USING LETTERS OF THE
          ALPHABET AND COMMAs"
110 INPUT LINE P$
120 PRINT P$
```

During RUN if you enter ANIL, SHARMA AND ANURAG ARE GOOD FRIENDS.

The output will be

```
ANIL     SHARMA AND ANURAG ARE GOOD FRIENDS.
```

ANIL will appear in the print zone first and SHARMA onwards will appear in the second print zone onward. Try this program yourself on the computer.

TAB Function

The TAB function positions text along a line. This can be used with PRINT as well as INPUT statements.

It has the following format:

```
PRINT TAB (C,R)
INPUT TAB (C,R)
PRINT TAB (C)
INPUT TAB (C)
```

The two parameters have numeric values C,R representing the column and row numbers of the screen. In different MODEs the number of the column and row is different. In MODE 7 there are 40 columns and 25 rows.

Consider the following statement:

```
50 PRINT TAB (10, 12) "KENDRIYA VIDYALAYA"
```

In the output of the above statement the first character K will appear in the 10th column and 12th row and E will appear in the 11th column of the same row, and so on.

EXERCISES

1. If A = 37, B = 25. Prepare a program by using LET and PRINT statements to calculate, (a) A + B, (b) A − B (c) A*B, (d) A/B.
2. Prepare a program by using the PRINT statement to see your name and address on the VDU.
3. Prepare a program to calculate the area of a trapezium by using the formula ((A + B)/2)*H, where A and B are the pair of parallel sides and H is the distance between them.
4. Modify Program 1 by using the INPUT statement to allow the operation to enter a radius of your own choice.
5. Prepare a program to calculate compound interest which gives the amount and CI for any desired value of P,R,T.
6. Consider the following program:
   10 PRINT "JANUARY", "FEBRUARY", "MARCH", "APRIL", "MAY", "JUNE", "JULY", "AUGUST".
   What would the output be?
7. Write a program in which the computer asks you which your favourite food is. The computer should reply 'I also like' (name of food given by you as operator).

8. Enter this program

```
10 PRINT "1419";
20 PRINT "327962";
30 PRINT "75";
40 PRINT "6";
50 PRINT "250";
```

RUN it and see the output.

9. Point out the error in the following statements:

```
[a] LET R=15*/3
[b] LET X=3*20; 15+2
[c] LET Z=30 + Y$
[d] PRINT "ANANDKUMAR"
[e] PRINT 'DEEPESH'
[f] PRINT "ENTER RADIUS"; R $
[g] INPUT "ENTER YOUR NAME";
[h] INPUT TAB (60, 100) "SHRINIVAS RAGHAVAN"
```

10 What are the outputs of the following programs?

```
[a] 10 LET A=20
    20 LET B=5
    30 LET C=A-B
    40 PRINT C
    50 PRINT A/B
    60 PRINT (B/A) *C
    70 END
[b] 10 PRINT "APPU KUMAR"
    20 PRINT " "SPORT SCHOOL"
    30 PRINT " "ASIAD NAGAR"
    40 END
[c] 10 LET A $="CHILDREN"
    20 LET B $="LIKE"
    30 LET C $="SWEETS."
    40 PRINT A $,B $,C $
    50 PRINT A $+B $+C $
    60 PRINT A $;B $;C $
    70 PRINT TAB (4,5) A $
    80 PRINT TAB (14,5) B $
    90 PRINT TAB (20,5) C $
   100 END
```

CHAPTER 6

Read, Data and Restore

The LET and INPUT statements have limitations. It is very cumbersome to write each LET statement or enter each INPUT statement. If the data is very large, then READ and DATA are included in the program well in advance. Whereas the INPUT statement allows the user to enter the data during the program, which consumes a great deal of execution time and reduces the speed of the computer, the READ statement has the following format:

READ A,B,C1, A$, K$

Where A,B,C1,A$ and K$ are variables separated by commas.

The variables listed in the READ statement have corresponding values in the DATA statement. The corresponding DATA statement of the above READ statement may be as follows:

DATA 5,9, -7, JOHN, ANANDKUMAR

Example

```
10 READ X,Y,Z,P $,M $,B $, A1
20 PRINT X,Y,Z
40 PRINT P $,M $,B $, A1
20 DATA 3,-9,27.6,PAWANSINGH,KAVITA,BOMBAY, 5.236
```

A DATA statement can be anywhere in the program but not after the END statement. Generally the DATA statement is placed just before the END statement so that it can be easily changed whenever it is necessary to do so.

There is a one to one correspondence between the type of the variables in the READ statement and their values in the DATA statement.

Try to run the following programs on your Micro and note the outputs.

Program 1

```
10 READ P,Q,R,B $,N1
20 PRINT P,Q,R,B$,N1
30 DATA 100, 2.525, 9786, NAGPUR, -125
40 END
```

The output will be as follows:

```
        100           2.525          9786          NAGPUR
       -125
```

Now change the above program by adding the following new lines:

```
20 PRINT P,Q,R
25 PRINT B$
```

Note down the output yourself.

Enter the following programs and note down the outputs:

Program 2

```
10 READ P
20 READ Q
30 READ R
40 READ B$
50 READ N1
55 PRINT B$,P,Q,R,N1
60 DATA 100, 2.5356
70 DATA 9786
80 DATA NAGPUR, -125
90 END
```

Program 3

```
10 DATA 2100, 42.5295, 9.786
20 DATA NEW DELHI, -125
30 READ A, B, C, D$, NUMBER
40 PRINT D$,A,B,C, NUMBER
50 END
```

RESTORE Statement

Sometimes it is very useful to have several sets of data in one program. For this the user can use the RESTORE statement which enables the data pointer to be moved from one set of data to another. The keyword RESTORE resets the data pointer to the first set of data in the program.

Program 4 To calculate the arithmetic mean.

```
 10 PRINT "ARITHMETIC MEAN"
 20 C=1
 30 SUM=0
 40 READ D
 50 IF D= -9999 THEN 100
 60 LET SUM=SUM+D
 70 LET C=C+1
 80 GOTO 40
 90 REM * CALCULATIONS OF NUMBER OF DATA *
100 LET N=C-1
```

```
110 REM * MEAN *
120 PRINT "AVERAGE OF";N;"OBERVATIONS."
130 PRINT ' ' "ARITHMETIC MEAN= "; SUM/N
140 DATA 50.9,90.8,45,76.8,87.8
150 DATA 65,72.5,54.7,12.6,76.4
160 DATA 55,27,35.8,71.6,30.5,-9999
200 END
```

RUN and see the output.

You can change the DATA in line 140, 150 and 160 as required.

Program 5

```
10 READ A,B,C$
20 PRINT "A= ";A: PRINT" B= "; B : PRINT "C$= ",C$
30 RESTORE
40 READ A1, B1
50 PRINT "A1 - B1 = ";A1 - B1
60 DATA 50,-75,RAHIM
70 END
```

Note the output carefully. Remove line 30 and RUN the program again and see the difference.

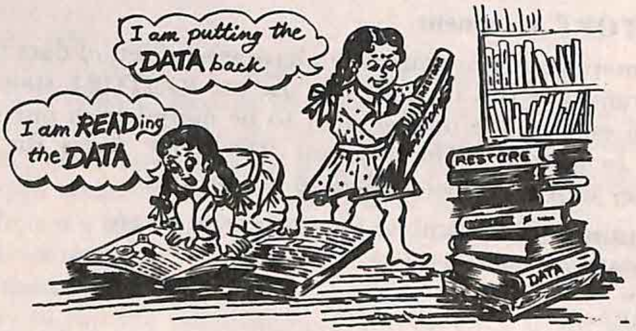

I am putting the DATA back
I am READing the DATA
RESTORE
DATA

Fig. 6.1

EXERCISES

1. Write a program to find the average of the marks obtained by the students of your class.
2. Write a program to print out the names of ten students.
3. Write a program to find the sum of twenty marks given by a friend of yours.
4. Find out the errors in the following programs and correct them so that they RUN properly.

```
[a] 10 DATA 25,9,GOPAL
    20 READ A$,B,C
    30 PRINT A$,B,C
    40 END
[b] 10 PRINT, A$,B,K$
    20 READ A$,B,K$
    30 DATA INDIA, 500, MOTHERLAND
[c] 10 READ A,B,C
    20 DATA 200,400
    30.PRINT A-B
    40 PRINT B-A
    50 PRINT (2*A-B)^0.5
    60 END
```

CHAPTER 7

Some More Hints on How to Write and Enter a Program

Multiple Statement in a Line

In the BBC Microcomputer, more than one statement can be placed in a single line. This saves time and computer-storage space. The statements are separated by a colon (:).

Example

```
10 LET A=20 : LET B=25 : LET C=2
20 LET D=(A-B) *C : LET E=(B-A)*C
30 PRINT D : PRINT E
```

CLS Statement

It would be better to use CLS in the first line of every program so that the computer executes its first task, which is to clear the screen of the monitor. The CLS statement means 'clear the screen'.

REM Statement

The REM statement stands for REMark. This provides an appropriate heading for the program. It is particularly useful in long and complicated programs. It cannot be read by the computer during execution but it appears in the LIST, so it helps when a user revises the program subsequently.

Program For CLS and REM statements

```
10 CLS : LET P= 1500 : LET R= 10 : LET T= 5
20 REM * MATHEMATICS *
30 A= P* ((1+(R/100))^T)
40 LET CI= A-P
50 REM * PRINTING THE RESULT*
600 PRINT "PRINCIPAL=";P
700 PRINT "RATE % =";R
80 PRINT "TIME= ";T;"YEARS"
90 PRINT "AMOUNT=Rs";A: PRINT "COMPOUND INTEREST= Rs "; CI
100 END
```

Replace line 10 by the INPUT statements that follow and see the difference.

```
10 CLS: INPUT "ENTER PRINCIPAL IN Rs.";P : PUT" ENTER RATE % "; R :
INPUT' ENTER TIME IN YEARS";T
```

Using a User Defined Key

If the user has to type a statement repeatedly, say PRINT in the same program. There is a provision for this in the BBC Micro.

Enter the following line before typing your program, and you will get this facility.

```
*KEY0 PRINT          press RETURN key
```

Now when you press f0 the word PRINT will appear on the monitor. Similarly, the user can define other keys in accordance with their needs. This will save time.

The user can enter more than one word into the User Defined Key,such as:

```
*KEY 2 SOUND 1, -15,53,20
```

Now let us assume that you have a program in MODE 2. If you want to see its LIST on the screen, you will find it very difficult to read the bold and wide characters in MODE 2. You have to give the following set of instructions:

```
MODE 7                        press RETURN
LIST                          press RETURN
```

But there is an easier equivalent instruction that can be programmed with the help of the User Defined Key as:

```
*KEY 4 MODE 7¦M LIST ¦M
```

The '¦M' symbol can be obtained by pressing SHIFT and the key marked '¦' near the BREAK Key.

In any program, you have to RUN it again and again. For this purpose you may program/define the User Defined Key. Let us take a key f5.

```
*KEY 5 RUM ¦M
```

Now whenever you press f5, your program will RUN. There is no need to press the RETURN key.

You can even program the BREAK, cursor control keys and COPY key. Refer to the USER GUIDE for further detail.

Renumbering the Line Statements of a Program

Sometimes you need to insert 20 lines between line number 40 to 50. It is quite difficult because there is only a 10 line space available between 40 and 50. For this purpose you have to RENUMBER the line numbers of your program by the key word RENUMBER.

The command RENUMBER will renumber a program, giving the first number 10, the second 20, the third 30, and so on.

The command RENUMBER 250 will renumber a program giving the first line 250, the second 260 and so on.

The command RENUMBER 150, 5 will renumber a program giving the first line number 150, the second 155, the third 160, and so on.

Do not forget to LIST after the command RENUMBER.

Procedure to Renumber

1. Prepare/LOAD a program which is to be renumbered.
2. Type RENUMBER (press RETURN)
3. Type LIST (press RETURN)

Sometimes it is necessary to list a few lines to revise a lengthy program. Let us take for example lines 40 to 90. For this purpose follow the following steps:

1. LOAD/prepare a program
2. Type LIST 40, 90. (press the RETURN key)

You will see all the lines from 40 to 90 on the screen.

MODE 6 provides an interesting way of listing programs.

Procedure:

1. LOAD/prepare a program
2. Type MODE 6 (press the RETURN key)
3. Type VDU 19,0,4,0,0,0 (press the RETURN key)
4. Type LIST (press the RETURN key)

This helps the user to read the listing clearly on the screen. You must try it.

A FEW BASIC STATEMENTS AND THEIR ABBREVIATIONS IN THE BBC MICROCOMPUTER

(The reader is advised to consult the USER GUIDE for greater detail)

BASIC KEY WORD	ABBREVIATION
AUTO	AU.
CHAIN	CH.
DATA	D.
GOTO	G.
INPUT	I.
NEXT	N.
OLD	O.
LOAD	LO.
LIST	L.

PRINT	P.
RENUMBER	REN.
RESTORE	RES.

Users are advised to use these abbreviations to save time.

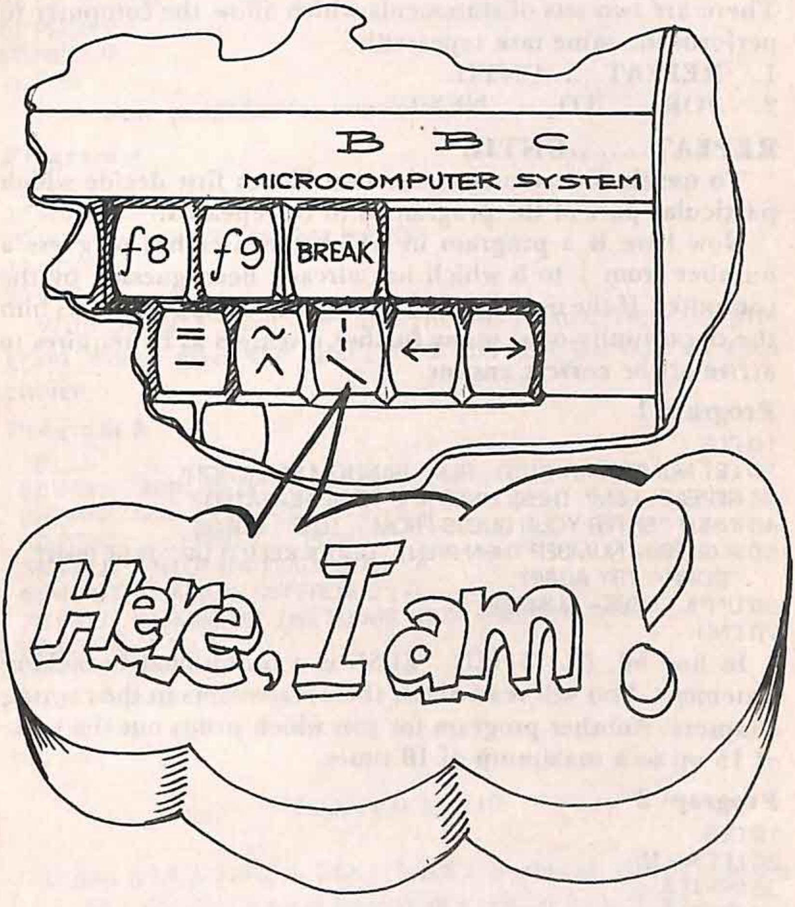


Fig. 7.1

CHAPTER 8

To Carry Out the Same Instructions

There are two sets of statements which allow the computer to perform the same task repeatedly.

1. REPEAT......UNTIL
2. FOR......TO......NEXT

REPEAT........UNTIL

To use these statements, the user has to first decide which particular part of the program is to be repeated.

Now here is a program in which the user has to guess a number from 1 to 5 which has already been guessed by the computer. If the user's guess is wrong, the computer allows him the opportunity of as many further attempts as he requires to arrive at the correct answer.

Program 1

```
10 CLS
20 LET NUMBER= RND (5) : REM* RANDOM SELECTION*
30 REPEAT : REM* THESE LINES ARE TO BE REPEATED *
40 INPUT "ENTER YOUR GUESS FROM 1 TO 5"; GUESS
50 IF GUESS= NUMBER THEN PRINT "GOOD! KEEP IT UP " ELSE PRINT
    "SORRY. TRY AGAIN"
60 UNTIL GUESS= NUMBER
70 END
```

In line 50, IF.. THEN.. ELSE is a conditional branching statement. You will read about these statements in the coming chapters. Another program for you which prints out the table of 15 up to a maximum of 10 times.

Program 2

```
10 CLS
20 LET A= 15
30 PRINT A
40 REPEAT
50 A= A+ 15
60 PRINT A
70 UNTIL A= 150
80 END
```

FOR.... TO.... NEXT Statements

The above program can be written with the FOR....TO..... NEXT statement which also does the same task as the REPEAT......UNTIL statement.

Program 3

```
10 CLS
20 LET A= 15
30 FOR M= 1 TO 10
40 LET P= A*M
50 PRINT P
60 NEXT M
70 END
```

One more program for you to try:

Program 4

```
10 CLS
20 FOR N= 1 TO 20
30 PRINT N; "HELLO! I AM BBC MICRO"
40 NEXT N
50 END
```

With this statement one can write a sequence. Here is a program which gives the ARITHMETIC SEQUENCE of your choice.

Program 5

```
 10 CLS
 20 PRINT "ARITHMETIC SEQUENCE"
 30 PRINT "= = = = = = = = = = = = = ="
 40 PRINT " " : PRINT " " : PRINT " "
 50 INPUT "ENTER THE FIRST TERM"; A
 60 INPUT "COMMON DIFFERENCE"; D
 70 INPUT "NUMBER OF THE TERMS TO BE PRINTED"; T
 80 FOR N= 1 TO T
 90 PRINT A; ",";
10 LET A= A+ D
110 NEXT N
120 END
```

NESTED LOOP

When a loop FOR... TO... NEXT is placed within another loop then the first loop is known as a nested loop. One can put one or more loop(s) inside another loop.

Insert line number 105 as under in the PROGRAM 5 and see the effect.

```
105 FOR W=1 TO 300: NEXT W : REM * WAITING TIME*
```

This line will allow the computer to count from 1 to 300. Here the computer will wait for approximately 3 seconds and then print the next term. So printing will appear slowly, term by term.

While nesting the loops, care should be taken to see that one loop lies completely inside the other loop. As in arithmetic and algebra where you use brackets, avoid overlapping of loops. Also, the outer and inner loops cannot have the same running variables.

Program 6

```
10 REM * NESTED FOR....NEXT LOOPS *
20 FOR P=1 TO 10
30 REM * OUTER LOOP IN LINE 20 *
40 FOR Q=1 TO P : REM INNER LOOP
50 PRINT "*";
60 NEXT Q
70 PRINT " ": REM * BLANK LINE *
80 NEXT P
90 END
```

There is another loop in lines 40 50 & 60. It is completely within the outer one.

FOR ..... TO ..... STEP ... NEXT Statement

The following program uses the FOR ... TO ... STEP ... NEXT statement and this will explain the purpose of the STEP with the FOR ... TO ... NEXT statement.

Program 7

```
10 CLS
20 FOR N=2 TO 3.2 STEP .1
30 PRINT "N", N^2", "N^3"
40 LET S=N^2
50 LET C=N^3
60 PRINT N,S,C
70 NEXT N
80 END
```

The output of this program will be the table of the square and cubes of the numbers from 2, 2.1, 2.2, 2.3, ... upto 3.2.

This tedious job of calculation is performed very easily and rapidly by the computer.

Now change line 20, as in the above program, as follows:

Fig. 8.1

```
20 FOR N=42.2 TO 43 STEP .05
```

and see the result.

EXERCISES

1. Rewrite PROGRAM 2 by using the FOR ... TO ... NEXT statement.
2. Prepare any program with the help of the REPEAT ... UNTIL statement.
3. What are the differences between REPEAT ... UNTIL and FOR ... TO ... NEXT statements?
4. Write a PROGRAM to find a number divisible by 15 and 21 in the 200 to 800 range.
5. Write a PROGRAM to calculate the sum of the first 25 even natural numbers in two different ways.
6. Explain each line of the following program:

```
 10 CLS
 20 PRINT TAB (10,37) "TABLE FROM 5 to 15"
 30 REM OUTER LOOP STARTS
 40 FOR M=5 TO 15
 50 PRINT TAB (5,5) "TABLE OF ";M
 60 REM INNER LOOP STARTS
 70 FOR N=1 to 10
 80 PRINT M*N
 90 NEXT N : REM INNER LOOP ENDS
100 INPUTTAB (3,25)"Press space bar to continue"; A$
110 NEXT M : REM OUTER LOOP ENDS
120 END
```

Now try this program on your microcomputer.

7. Correct the errors in the following program:

```
[a] 10 FOR M=1 TO = 10
    20 PRINT "GOOD MORNING"
    30 NEXT K
[b] 10 FOR N=10 TO 2 STEP 2
    20 PRINT N
    30 NEXT N
[c] 10 FOR K=5 TO 20
    20 FOR P=1 TO 10
    30 PRINT K*P
    40 FOR W=1 TO 1000 : NEXT W
    50 NEXT K
    60 NEXT P
```

8. Prepare a program to find the sum of natural numbers from 1 to a given natural number by using
   - [i] FOR ... TO ... NEXT
   - [ii] REPEAT ... UNTIL

9. In the following FOR ... TO ... statements, write the value taken by the index variable

FOR .. TO .. statements	Index variable	Values
[i] 20 FOR A=1 TO 7 STEP 2	A	1,3,5,7
[ii] 130 FOR K=10 TO 3 STEP−2	________	________
[iii] 40 FOR X=245 TO 950 STEP 200	________	________
[iv] 70 FOR Y=1.2 TO 2.4 STEP 0.4	________	________
[v] 205 FOR delay=1 TO 3	________	________
[vi] 32 FOR H9=5 TO 25 STEP 5	________	________

10. Explain each of the statements in PROGRAM 2.

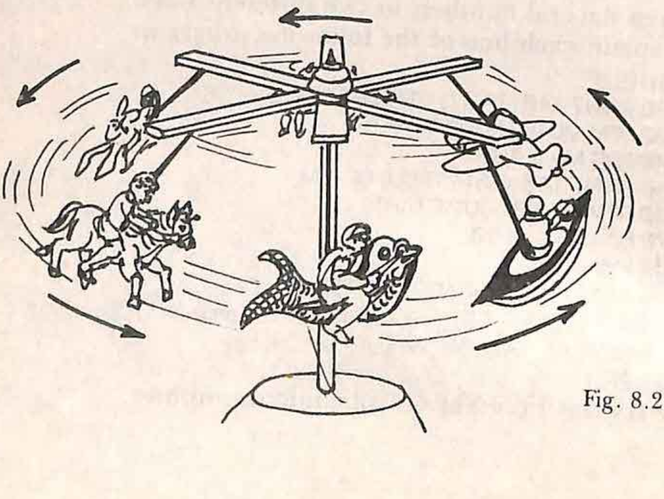


Fig. 8.2

CHAPTER 9

Jumping and Branching Statements

In a computer program the statements are generally executed sequentially in the increasing order of line numbers of the statements. But sometimes it is necessary to repeat the same set of instructions until a certain specific condition is satisfied. In such cases, the order of execution is controlled by jumping (forward or backward), branching and looping. You have already studied the looping statements, i.e. FOR .... TO .... NEXT and REPEAT .... UNTIL.

Now for the jumping and branching control one should use GOTO n, ON N GOTO n, GOSUB and ON N GOSUB.

GOTO n Statement

This is one of the control statements which causes the computer to jump from one line number to another. Its format is GOTO n.

Program 1 The following program prints out the multiple of 8 and performs a backward jump.

```
10 CLS
20 LET A=8
30 PRINT A; " ";
40 LET A=A+8
50 GOTO 30 : REM * BACKWARD JUMP*
60 END
```

This program is endless. To come out of the program press the ESCAPE key.

Program 2 The following program illustrates the forward jump.

```
10 CLS
20 LET A=200
30 LET B=50
40 GOTO 70 : REM * FORWARD JUMP *
50 LET A= A * A : REM * New value of A= 40000 *
60 LET B= A * B : REM * New value of B= 10000 *
70 LET P=A-B
```

```
 80 LET Q=A/B
 90 PRINT P,Q,A,B
100 END
```

The output of this program will be

```
        150           4             200              50
```

Now remove line 40 from it and the output will become

```
      30000           4           40000           10000
```

Thus we see the program with line 40 will omit the execution of line 50 and 60. And the program without line 40 will execute line 50 onwards.

Now slightly change the above program by introducing the following lines.

```
40 GOTO 95
95 PRINT "GOOD BYE ! MR. USER"
```

```
  The output will be
  GOOD BYE ! MR. USER
```

In this way it is clear that the control jumps from line 40 to line 95. Here the computer is not executing the section of the program between lines 40 and 95.

ON N GOTO ... Statements

These statements are used to transfer control to one of several lines of the program, depending on the value of a numeric expression. Its format is ON N GOTO n1,n2,n3 ... Here N is used for option number and n1,n2 ... used for line numbers.

Program 3 An option part of a program.

```
10 CLS
20 PRINT TAB(10) "O P T I O N"
30 PRINT TAB(10) "= = = = = = = = = ="
40 PRINT TAB(6,5) "CIRCLE.......... 1"
50 PRINT TAB(6,7) "SPHERE........ 2"
60 PRINT TAB(6,9) "CONE........... 3"
70 PRINT TAB(6,11) "END........... 4"
80 INPUT TAB(4,13) "ENTER THE OPTION"; N
90 ON N GOTO 1000, 2000, 3000, 4000
.
.
.
1000 CLS : PRINT TAB(10) "CIRCLE"
.
.
1999 GOTO 10
```

```
2000 CLS : PRINT TAB (10) "SPHERE"
.
.
2999 GOTO 10
3000 CLS : PRINT TAB (10) "CONE"
.
.
4000 END
```

If N = 1, then control is transferred to line number 2000 which is the starting line of the CIRCLE program. If N = 2,3,4 then control is transferred to SPHERE, CONE and END, respectively. See flow chart on page 60.

GOSUB Statement

A GOSUB statement is used to take the computer control to the n line number from the program's mainstream. Its format is

GOSUB n where n is a line number.

A RETURN must be used at the end of subroutine's execution to return the computer control from the subroutine to the main program. Do not use a GOTO statement to come out of the subroutines.

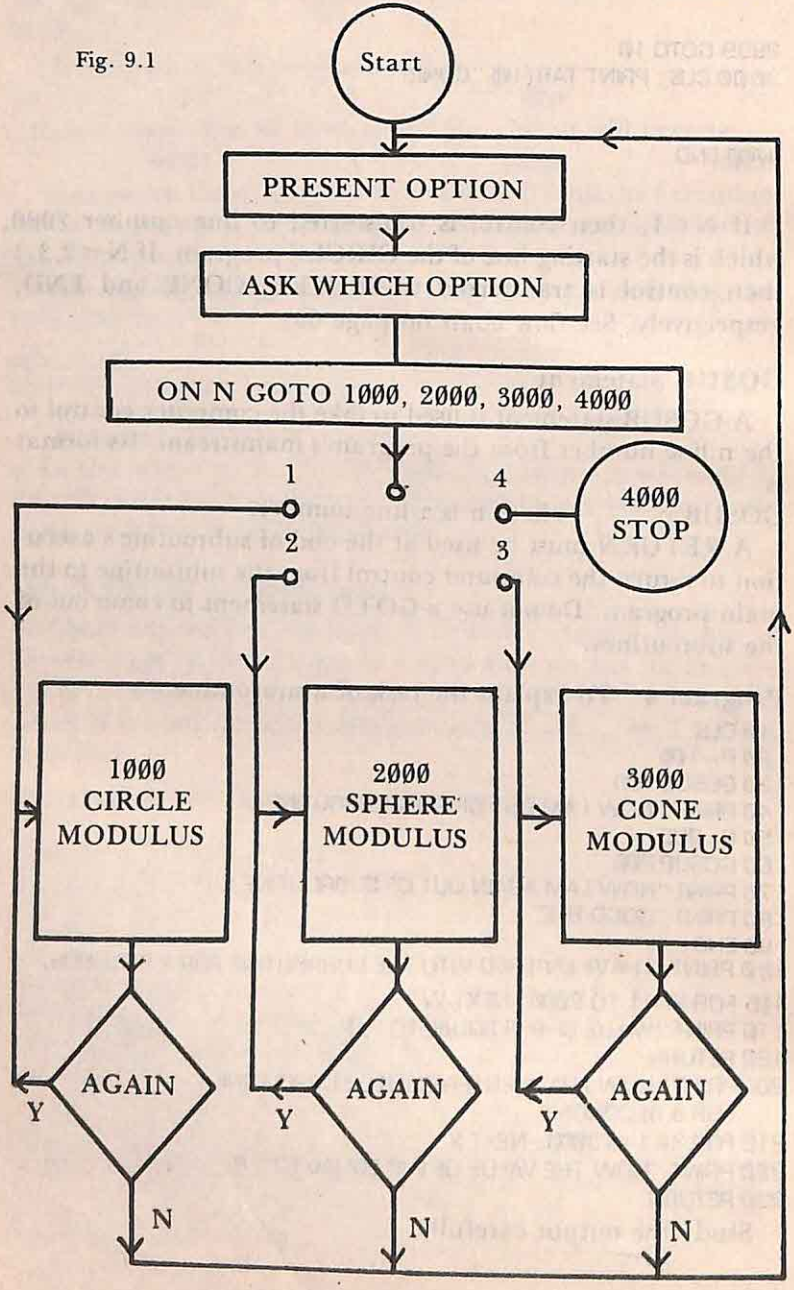
Program 4 To explain the task of a subroutine

```
 10 CLS
 20 P= 100
 30 GOSUB 100
 40 PRINT "NOW I AM OUT OF THE SUBROUTINE I."
 50 P= 250
 60 GOSUB 200
 70 PRINT "NOW I AM AGAIN OUT OF SUBROUTINE II."
 80 PRINT "GOOD BYE'
 90 END
100 PRINT "I HAVE ENTERED INTO THE SUBROUTINE FOR 2 SECONDS";
105 FOR W=1 TO 2000 : NEXT W
110 PRINT "VALUE OF P IS EQUAL TO "; P
120 RETURN
200 PRINT "NOW I HAVE ENTERED INTO SUBROUTINE II
    FOR 3 SECONDS."
210 FOR X= 1 to 3000 : NEXT X
220 PRINT "NOW THE VALUE OF P IS EQUAL TO"; P
230 RETURN
```

Study the output carefully.

FLOW CHART TO SHOW ON...GOTO...STATEMENT

Fig. 9.1



ON N GOSUB Statement

It is similar to the ON N GOTO statement except that the control in ON N GOTO returns to the main program by GOTO whereas the control in GOSUB returns by the RETURN statement (not by RETURN key), as you have seen in PROGRAM 4.

There is one alternative to the statement in line 90 in PROGRAM 3, which is as follows:

```
90 ON N GOSUB 1000,2000,3000,4000
```

In this case the control is passed to the subroutines indicated and then returned to the next line.

A part of a program containing the GOSUB statement:

```
100 GOSUB 1000
110 PRINT "VOLUME" ; V

990 RETURN
1000 PRINT "CONE"
1010 INPUT "ENTER THE RADIUS AND HEIGHT OF A CONE"; R,H
1020 LET V= PI * (R^2) * H

:
:
1500 RETURN
```

Explanation

100 the control transfers to line 1000. Then the computer executes the line numbers from 1000 to 1500 in accordance with the statements given in these lines. In line number 1500 the control returns to line number 110, which is the first statement after GOSUB, and then the computer proceeds till the end of the program is reached.

Some other uses of ON ... GOTO and ON ... GOSUB are given in the following examples:

```
20 ON ERROR GOTO 5000
60 ON ERROR PRINT "THE COMPUTER HAS NOT FOLLOWED YOU."
30 ON ERROR GOSUB 9000
50 ON ERROR OFF : REM * Without GOTO AND GOSUB *
```

The user is advised to use these statements in his program.

EXERCISES

1. Can a subroutine contain more than one RETURN statement? Explain.

2. Prepare a program in which backward and forward jumping are involved.
3. Can a control be transferred out of a subroutine by means of a GOTO statement?
4. What are the main differences between ON ... GOTO and ON ... GOSUB statements?
5. Prepare a program to solve two/three simultaneous linear equations along with a quadratic equation by using subroutines.
6. Point out the errors and correct the following statements:

```
[i] 130 GOTO 60,80,90,230
[ii] 200 ON P GOTO 180,30,200,300
[iii] 40 ON Z GOTO 50,100,100,150,200
[iv] 120 ON K GOTO 100;20;30;400
```

7. Locate the error(s) in the following program;

```
50 GOSUB 205
.....
205 A=B+C+D
.....
230 GOTO 10
280 RETURN
290 RETURN 10
```

8. The abbreviations of the statements are listed below:

Statement	*Abbreviation*
GOTO	G.
GOSUB	GOS.
RETURN	R.

Prepare a program by using the above abbreviations. Does it work?

CHAPTER 10

Conditional Branching

IF ... THEN Statement

The IF ... THEN statement is used to carry out conditional branching in which control will be transferred to the next statement; otherwise, the statement following the IF ... THEN line number will be executed next.

Fig. 10.1

Program 1 A program for the geometrical sequence (G.P.).

```
10 CLS
20 LET A=5 : REM * 5 is the first term *
30 LET N=2 : REM * 2 is the common ratio*
40 PRINT A ; ",";
50 LET A=A*N
60 IF A <= 320 THEN 40
70 PRINT "........"
80 END
```

The output of the program will be

```
5,10,20,40,80,160,320,.........
```

Line 60 may be written as

```
60 IF A <= 320 THEN GOTO 40
```

It is to be noted here that GOTO in line 60 is optional.

The following is a part of the program where the computer asks the user to restart the program.

```
300 INPUT "DO YOU WANT ANOTHER RUN" ; A$
310 IF A$ = "YES" THEN 10
320 IF A$ = "NO" THEN 340
330 PRINT "REPLY UNACCEPTABLE" : GOTO 300
340 END
```

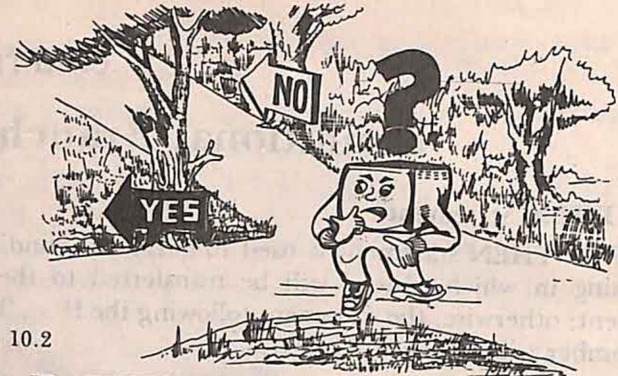


Fig. 10.2

The IF...THEN statement can also be used for decision-making. Let us try the following program to solve a quadratic equation.

Program 2

```
 10 CLS : PRINT TAB (10) "QUADRATIC EQUATION"
 20 INPUT "ENTER COEFFICIENT OF X^2" ; A
 30 INPUT "ENTER COEFFICIENT OF X" ; B
 40 INPUT "ENTER CONSTANT TERM"; C
 50 LET D= B^2 - 4*A*C
 60 IF D < 0 THEN PRINT "NO REAL ROOTS" : GOTO 20
 70 IF D > =0 THEN PRINT "REAL ROOTS"
 80 LET X= (-B+ SQR(D))/2*A
 90 LET X2 (-B- SQR(D))/2*A
100 PRINT "X1 = ";X1
110 PRINT "X2= "; X2
120 END
```

In line 70 after THEN the line number is not given but printing work is assigned to the computer.

IF ... THEN is also used with more complicated conditions involving the statements NOT, OR and AND ; for example:
20 IF NOT (X = 10) THEN PRINT "X NOT 10"
Now another use of the IF ... THEN statement with an OR condition.

Program 3

```
10 INPUT "ENTER RADIUS & HEIGHT OF A RIGHT CYLINDER" ; R ; H
20 IF R< 0 OR H< 0THEN PRINT "NOT POSSIBLE" : GOTO 10
30 LET V= PI * (R^2)*H
40 PRINT "VOLUME= " ; V
50 END
```

Here is a program which prints a number divisible by 13 and 23 within the range 1 to 1000. The IF ... THEN statement includes the AND condition.

Program 4

```
100 LET D1 = 13 : LET D2 = 23 
110 FOR N=1 TO 1000
120 REM * DIVISION TRIAL *
130 LET K1 = N/D1 : LET K2 = N/D2 : REM * LET is optional in BBC *
140 IF INT (K1)= K1 AND INT (K2)= K2 THEN PRINT N
150 NEXT N
160 END
```

The output will be 299,598 and 897

There is one more IF ... THEN ... ELSE conditional statement for example:

```
1210 IF K= S THEN PRINT "EQUAL" ELSE PRINT "NOT EQUAL"
```

Program 5

```
10 CLS
20 PRINT TAB (1,5) "DO YOU WANT ADMISSION TO THE SCHOOL"
30 INPUT P$
40 IF P$= "NO" THEN 60 ELSE GOTO 70
60 CLS: PRINT TAB (1,5) "ITS GLAD TO KNOW THAT" : GOTO 90
70 CLS: PRINT TAB (1,5) CHR $ (141) "WELCOME"
80 PRINT TAB (1,6) CHR $ (141) "WELCOME"
90 END
```

EXERCISES

1. Write a program to find the sum of the integers 1 to 100.
2. What is the output of the following program?

```
10 LET A= 3
20 IF A= 5 OR A= 9 THEN 40
30 PRINT A
40 A= A+ 1
50 IF A < 25 THEN 20
60 END
```

3. The following program is to print out in reverse order the numbers from 15 to 1.

```
10 FOR N= 15 TO 1 STEP -1
20 PRINT N
30 NEXT N
40 END
```

Rewrite the above program, using the IF ... THEN statement.

4. Some IF ... THEN statements are given below. Which statements are incorrect and why?

```
A] 40 IF > =100 THEN 60
B] 20 IF N^3 > =400 GOTO 80
C] 75 IF P< > 10 THEN Q
D] 125 GOTO 20 IF R=5
E] IF A $="YES" THEN END
F] IF X+Y< =T THEN 90
G] IF B< =A/2 THEN PRINT "LESS OR EQUAL"
```

5. Analyse each line of the following program:

```
10 CLS : REM * DIVISIBILITY OF 5*
20 FOR N=5 TO 1000
30 LET D=N/5
40 IF INT(D)=D THEN PRINT N "DIVISIBLE BY 5" ELSE PRINT N" IS
    NOT DIVISIBLE BY 5" : FOR wait=1 TO 500 : NEXT wait
50 NEXT N
60 END
```

Test this program on your Micro.

6. Prepare a program to test general knowledge.
7. Explain each statement of the following program:

```
10 CLS: PRINT "SIR, THIS IS A AGRA HOTEL 10 CLS: COUNTER
    SERVICE"
20 INPUT "DO YOU WANT A ROOM";Y$
30 IF Y $="Y" THEN PRINT "SINGLE BED OR DOUBLE BED" :
    GOTO 100
40 IF Y $="N" THEN GOTO 200
100 INPUT A$
110 IF A $="SINGLE" THEN PRINT "NO SINGLE BED'S ROOM IS
     AVAILABLE "ELSE PRINT" YES, WE HAVE A DOUBLE BED'S
     ROOM FOR YOU SIR" : GOTO 300
200 PRINT "WHAT CAN I DO FOR YOU, SIR"
300 END
```

Enter this program and try to RUN.

8. RUN PROGRAM 5. Study the output of lines 70 and 80.

CHAPTER 11

Counting and Splitting the Characters

LEN and MID$ Statements

The LEN and MID$ statements are used to count and split the characters of a word/sentence respectively.

LEN: To count the number of the characters in a word or sentence.

Program 1 A program to count the number of characters

```
10 CLS: REM * CLEAR THE SCREEN*
20 PRINT "ENTER A WORD, COMPUTER WILL COUNT THE NUMBER OF
   THE CHARACTERS IN IT"
30 INPUT A$
40 LET C= LEN (A$)
50 PRINT "THE NUMBER OF THE CHARACTERS IN THE GIVEN WORD IS=
   ";C
60 END
```

Explanation of line 40: the computer will count the number of characters in the string A$ and assign the numeric value to the variable C.

MID$: To copy part of one string into another string.

Program 2

```
10 NAME $= "DEEPAKKUMAR"
20 B $= MID $(NAME$,5,3)
30 C $= MID $(NAME$,2,6)
40 D $= MID $ (NAME$, 10, 2)
50 PRINT B $,C $,D $
60 END
```

Explanation

Line number	Description
10	clear the screen
20	B$ will be the set of 3 characters from the 5th character onwards in NAME$ string.
30	C$ will be the set of 6 characters from 2nd character onwards in NAME$ string.
40	D$ will be the set of 2 characters from the

	tenth onwards in NAME$ string
50	Print out the B$, C$ and D$
60	End

RUN this program and you will see the following output:

```
      AKK              EEPAKK              AR
```

LEFT$ AND RIGHT$

These two statements are also used to split a word from left and right respectively. Its format is

LEFT$(< string variable >, < number >)
RIGHT$(< string variable >, < number >)

Program 3

```
10 A $= "RAJIVGANDHI"
20 B $= LEFT $ (A $,5)
30 C $= RIGHT $ (A $,6)
40 PRINT C $
50 PRINT B $
60 END
```

The output of the program will be

```
GANDHI
RAJIV
```

COUNT Statement

It is a statement which counts all the characters printed, using the PRINT statement.

Program 4 A program to print out exactly 200 'X'

```
10 CLS
20 A= 200
30 REPEAT PRINT "X"
40 UNTIL COUNT= A
```

Let us try this program on your Micro.

EXERCISES

1. Write a program which can print your name in the reverse order of letters.
2. Explain each of the lines in the following program:

```
10 CLS
20 PRINT "ENTER THE TWO INITIALS OF YOUR NAME"
30 PRINT "THEN LEAVE SPACE"
40 PRINT "ENTER YOUR SURNAME"
50 INPUT ; N $
60 B= LEN (N $)
```

```
70 FOR N = 1 TO B
80 X $ = LEFT $(N $,3)
90 Y $ = RIGHT (N$,B - 3)
100 Z $ = X $ + " " + Y $ : REM * ADDING STRING *
120 END
```

Now enter this program and RUN. Note down the output.

3. Write down the uses of the following statements and prepare

`MID $, LEFT $, COUNT, RIGHT $ and LEN`

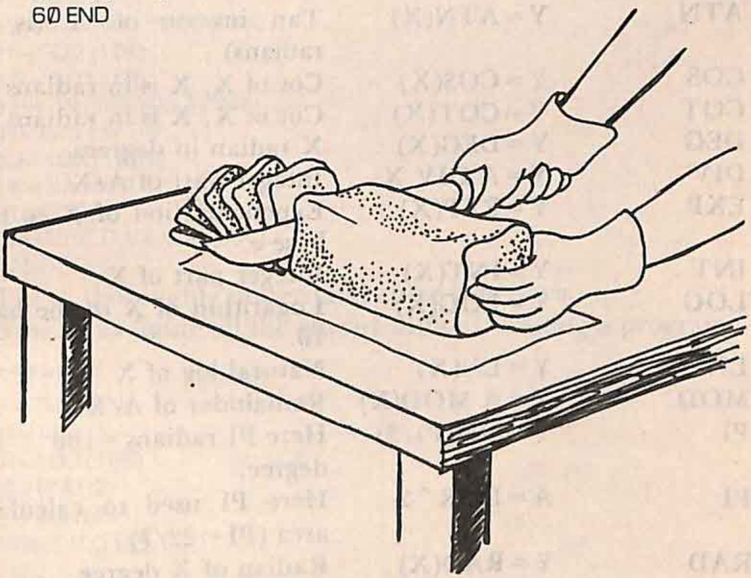
4. Replace lines 20 & 30 in PROGRAM 4 as follows:

```
20 A = 20
30 REPEAT PRINT "INDIA";
```

Note the difference between this new program and PROGRAM 4. Give a reason for this difference.

5. Correct the error(s) in the following program, if any:

```
10 LET A = 20
20 LET B = A/10
30 LET X $ = HINDUSTAN
40 X = LEN(X $)
50 P., (A - X) ^ 2
60 END
```



CHAPTER 12

Built-in Library Functions for Mathematical Calculations

The BBC microcomputer has some built-in library functions to facilitate mathematical calculations. Some of them are listed here.

SOME SPECIAL MATHEMATICAL STATEMENTS

Functions	Format	To Calculate
ABS	Y = ABS(X)	Absolute value of X
ACS	Y = ACS(X)	Cosine inverse of X (Y is in radians)
ASN	Y = ASN(X)	Sine inverse of X (Y is in radians).
ATN	Y = ATN(X)	Tan inverse of X (is in radians)
COS	Y = COS(X)	Cos of X. X is in radians
COT	Y = COT(X)	Cot of X, X is in radians
DEG	Y = DEG(X)	X radian in degrees
DIV	Y = A DIV X	Integer part of A/X
EXP	Y = EXP(X)	Exponentiation of X to the base e.
INT	Y = INT(X)	Integer part of X
LOG	Y = LOG(X)	Logarithm of X to the base 10.
LN	Y = LN(X)	Natural log of X
MOD	Y = A MOD(X)	Remainder of A/X
PI	Y = SIN(PI/3)	Here PI radians = 180 degree.
PI	A = PI*R^2	Here PI used to calculate area (PI = 22/7)
RAD	Y = RAD(X)	Radian of X degree
RND	Y = RND(X)	Random value of a natural X.
SGN	Y = SGN(X)	Signum function of X. SGN(X) = ABS(X)/X for all

		values of X< > 0
SIN	Y = SIN(X)	Sine of X. (X is in radians)
SQR	Y = SQR(X)	(Positive) square root of X.
TAN	Y = TAN(X)	Tangent of X. (X is in radians)

In all the above functions, the value of X is real. In the case of the trigonometric function, X is always taken in radians, whereas in case of inverse trigonometric functions the value of Y is always calculated in radians.

Now let us try out the following statements, one at a time, and note down their outputs carefully.

Enter a line, press RETURN and then RUN.

```
10 A= ABS (-25) : PRINT A
20 B= ABS (2563.67) : PRINT B
```

The output of the above line statements will be 25 & 2563.67.

Now try these programs:

Program 1

```
 10 C= RAD (57)
 20 D= SIN (PI/2)
 30 E= SIN (PI/4) *COS (PI/4)
 40 F= SQR (100)
 50 G= DEG (2PI)
 60 H=  INT (56308.1234)
 70 I= 425 DIV 30
 80 J= LOG (1000)
 90 K= 27 MOD 12
100 L= RND(2)
110 PRINT C,D,E,F,G,H,I,J,K,L
120 END
```

Try to thoroughly understand these functions.

Now let us figure all the output without running a program.

Program 2

```
10 P= SIN (PI/4)
20 Q= ASN (1)
30 R= LOG (100)
40 S= RND (2)
50 T= SQR (441)
60 PRINT P,Q,R,S,T
70 END
```

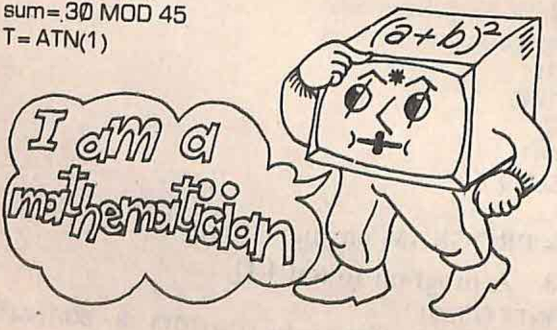
Run this PROGRAM on your Micro.

Program 3 A program to test I.Q.

```
10 CLS: PRINT "IQ TEST"
20 LET P= SIN (PI/2) : Q= ASN (1) : R= LOG (100) : S= SQR (44100) :
    T= DEG (PI/2)
 30 INPUT "SIN (PI/2)= ";A
 40 IF A= P THEN PRINT "CORRECT" ELSE PRINT "WRONG"
 50 INPUT "SIN INVERSE OF 1 = " ;B
 60 IF B= Q THEN PRINT "CORRECT" ELSE PRINT "WRONG"
 70 INPUT "LOG OF 100= "; C
 80 IF C= R THEN PRINT "CORRECT" ELSE PRINT "WRONG"
 90 INPUT "SQUARE ROOT OF 44100"; D
100 IF D= S THEN PRINT "CORRECT" ELSE PRINT "WRONG"
110 INPUT "DEGREE OF PI/2 RADIANS= ";E
120 IF E= T THEN PRINT "CORRECT" ELSE PRINT "WRONG"
130 END
```

EXERCISES

1. What are the purposes of ASN, RND and sin functions?
2. Write a program on guessing numbers from 1 to 5 by using the RND function.
3. Prepare a program to calculate the area and volume of a cylinder.
4. Write a program to calculate simple interest upto two decimal places.
5. Explain the statements given in PROGRAM 1 in writing.
6. Rewrite PROGRAM 3 so that the computer represents the user's score.
7. Point out the error(s), if any, and correct them in the following statements.
    - (i) `Y = SQR (- 25)`
    - (ii) `K$ = SIN (PI/2)`
    - (iii) `Z = 25 DIV 50`
    - (iv) `Number= RND (10)`
    - (v) `Value= sin (abs (- 25/50)) * COS (PI* 0)`
    - (vi) `I= ASN (SIN(pi))`
    - (vii) `sum= 30 MOD 45`
    - (viii) `T= ATN(1)`



CHAPTER 13

Dimensions and Defined Functions

The importance of a computer increases when large amounts of data are to be conveniently processed. The LET, READ, PRINT and INPUT statements are not very convenient in handling a large amount of data. Let us take the example of a bank which has a list of cheques. Suppose the amounts of the cheques are 2500, 1200, 350, 780, 950, 3500, 65000..., etc.; each element of this data can be named in the same order as A(1) = 2500, A(2) = 1200, A(3) = 350, A(4) = 780, A(5) = 950, A(6) = 3500, A(7) = 65000..., etc.

Thus you can have variables A(1), A(2), A(3)... and so on and in general, A(I). I is called the subscript and A(I) is called a single subscripted variable. The value of I can be assigned by the FOR ... TO ... NEXT loop. This makes it very easy to handle a large amount of data.

Subscripted string variables can also be used. Let us take a list of teachers who teach Class XI in your school. Suppose the list reads: Mrs DASGUPTA, Mr SUNIL, Mr GHULAMNABI, Mr MANOHAR and Mr HARINARAYAN. Then each element of this array can be named in the same order as: A\$(1) = Mrs DASGUPTA, A\$(2) = Mr SUNIL, A\$(3) = Mr GHULAMNABI ..., etc.

Let us take a program which involves the above subscripted string variables:

Program 1

```
 5 DIM N $ (10)
10 PRINT "THE COMPUTER READS THE SUBSCRIPTED STRING VARIABLES"
20 FOR I= 1 TO 5
30 READ N $ (I)
40 PRINT N $ (I)
50 PRINT " * THIS WAS NAME"; I : PRINT " "
60 FOR delay= 1 TO 1000: NEXT delay
70 NEXT I
80 DATA Mrs DASGUPTA, Mr SUNIL, Mr GHULAMNABI, Mr MANOHAR,
   Mr HARINARAYAN
90 END
```

The output will be as follows:

```
Mrs DASGUPTA
* THIS WAS NAME 1

Mr SUNIL
* THIS WAS NAME 2

Mr GHULAMNABI
* THIS WAS NAME 3

Mr MANOHAR
* THIS WAS NAME 4

Mr HARINARAYAN
* THIS WAS NAME 5
```

You usually come across a X1 + X2 + X3 + X4 + ... + X10 type of expression in mathematics. To solve this, see the following program.

Program 2 To calculate the arithmetic mean.

```
10 LET S=0 : DIM X (20)
20 FOR N=1 TO 10
30 READ X (N)
40 LET S=S+X (N)
50 NEXT N
60 LET M=S/N
70 PRINT "THE ARITHMETIC MEAN= ";M
80 DATA 50,27,80,20,19,13,60,100,15,81
90 END
```

The user can get the same result if he/she is not using a subscripted variable X(N) in line 30 and 40. But he cannot extend the program without the subscripted variable to calculate the standard deviations. The value of READ in statement 30 cannot be used again without the subscripted variable in the other parts of the same program.

Let us examine the following program carefully and try to RUN it on the computer.

Program 3 To calculate the mean and standard deviation.

```
 5 DIM X (10), Y (10)
10 FOR I=1 TO 10
20 READ X (I)
30 NEXT I
40 S=0
50 FOR J=1 TO 10
60 LET S=S+Y (J)
70 NEXT J
80 M=S/10
90 LET D=0
```

```
100 K=1 TO 10
110 D=D+(Y(K)-M)^2: REM* SUM OF THE SQUARE OF THE DEVIATIONS.
120 NEXT K
130 SD=SQR(D/10)
140 PRINT "THE MEAN IS"; M
150 PRINT "THE STD DEVIATION IS"; SD
160 DATA 20,15,25,30,28,12,10,18,24
170 END
```

Explanation

Line Number	
5	Store 10 locations each for X and Y
10,20,30	Read the data and assign the values
50,60,70	Find the sum of Y(1), Y(2), upto Y(10)
80	Calculate the mean
100,110,120	Calculate the mean of the squares of the deviations from the mean
130	Calculate the standard deviation
140,150	Print the result
160	Data
170	End

DIM Statement

Normally the computer reserves 10 storage locations for each variable name, but if more than 10 values are to be subscripted to a variable then the computer is to be directed to do this by the key word DIM, which stands for the word DIMensions. DIM is used to tell the computer to open enough memory locations to accommodate each element of a subscripted variable. Its format is

DIM < variable name > (assigned integer number)

Example

DIM A$ (16)	Sixteen locations for string variables
DIM N(20)	Twenty locations for numeric variables

The following program is to find out the total pocket expenditure of the 20 students of a class in a day.

Program 4

```
10 DIM S (20)
20 LET T=0
30 FOR N=1 TO 10
40 READ S (N)
50 LET T=T+S (N)
60 NEXT N
```

```
 70 PRINT "TOTAL EXPENDITURE IN Rs";T
 80 DATA 8,5,3,2,0,6,0,2,9,8
 90 DATA 6,3,8,5,4,7,0,1,2,3
100 END
```

Run the above program.

The following program is to calculate the expenditure of 10 students in a week. In this program different variables and subscripts for a single DIM statement, viz. variable name D for days, S for students and T for total, are involved. In DATA the expenditure is given in Rs.

Program 5

```
 10 DIM D (7), S (10), T (10), X (10)
 20 FOR N=1 TO 10
 30 LET T (N)=0
 40 NEXT N
 50 FOR D=1 TO 7
 60 FOR S=1 TO 10
 70 READ X (S)
 80 LET T (S)=T (S)+X (S)
 90 NEXT S
100 NEXT D
110 FOR S=1 TO 10
120 PRINT T (S); : PRINT " ";
130 NEXT S
140 DATA 2,4,5,0,6,0,7,8,4,9
150 DATA 9,7,8,9,5,7,6,0,8,3
160 DATA 4,6,8,7,9,7,6,5,7,0
170 DATA 6,7,5,9,3,0,9,8,7,4
180 DATA 3,9,7,0,5,4,3,2,1,8
190 DATA 7,6,9,4,0,4,3,2,5,6
200 DATA 0,6,9,7,5,3,1,0,3,0
210 END
```

Enter the above program and note down the output.

Program 6

```
10 DIM A (15)
20 REM * SUBSCRIPTED VARIABLES *
30 FOR I=1 TO 15
40 LET A (I)=2 * I
50 PRINT A (I)
60 NEXT I
70 LET T=A (5) + A (10) + A (15)
80 PRINT "Sum of the 5th, the 10th and the 15th subscripted data is ";T
90 END
```

There are five classes in a school. Each class has four sections of strengths given in the following table.

CLASS	A	B	C	D
XII	30	40	29	32
XI	40	36	38	30
X	37	35	34	32
IX	32	39	33	29
VIII	41	41	43	28

The following program declares a two-dimensional array and PRINTs it out in a tabular form.

Program 7

```
 10 DIM T(5,4) : REM * 2-DIMENSION ARRAY*
 20 FOR N= 1 TO 5
 30 FOR M= 1 TO 4
 40 READ T (N,M)
 50 PRINT T (N,M):
 60 NEXT M
 70 NEXT N
 80 DATA 30,40,29,32
 90 DATA 40,36,38,30
100 DATA 37,35,34,32
110 DATA 32,39,33,29
120 DATA 42,41,43,28
130 END
```

RUN this program and then modify it so that it also prints classes.

Can you guess the output of the following program? Run the program and verify it yourself.

Program 8

```
 10 DIM A $ (4,5) : REM * 2-DIMENSION ARRAY *
 20 REM * READING THE DATA *
 30 FOR I= 1 TO 4
 40 FOR J= 1 TO 5
 50 READ A $ (I,J)
 60 NEXT I
 70 NEXT J
 80 REM * PRINTING *
 90 FOR M= 1 TO 4
100 FOR N= 1 TO 5
110 REM * PRINTING *
120 PRINT A $ (M,N)
130 NEXT N
140 PRINT " "
150 NEXT M
160 DATA AB,BC,CD,DE,EF,FG,GH,HI,IJ,JK,KL,LM
170 DATA MN, NO,OP,PQ,QR,RS,ST,TU,
180 END
```

User Defined Function

This function is used to create any new function (other than those given in Chapter 12) for easy programming. The key word DEF provides an easy way of achieving this. Its format is DEF FNa(b) = Expression .

Where a stands for the name of a function denoted by a letter of the alphabet.

b stands for the argument of the function which is a variable in the expression.

Examples of some of the user defined functions are:

```
(i) DEF FN A (X)= X^2 - 2*X + 7
(ii) DEF FN B (X,Y)= X^2 + 2*X*Y + Y^2
(iii) DEF FN C (P,Q)= SQR (ABS (P - Q))
(iv) DEF FN X (P,R,N)= P* (1 - R/100)^N
(v) DEF FN A (P $,Y)= RIGHT $ (P $,Y)
```

The variables used in the bracket on the left hand side must be of the same type as used on the right hand side.

DEF FN A(X) = B^2 − 34*B is clearly wrong. The correct way of using the above function is

DEF FN A(X) = X^2 − 34*X

or, DEF FN A(B) = B^2 − 34*B.

Here you are given some programs based on the DEF function statements. You should try them on your computer and then prepare some other programmes of a similar kind.

Program 9 To calculate the escape velocity of a rocket.

```
10 LET G= 6.67*10^(-11): REM gravitational constant
20 LET M= 6.0*10^24: REM mass of the earth
30 LET R= 6.4*10^6: REM radius of the earth
40 DEF FNV (M,R )= SQR (2*G*M/R)
500 PRINT "ESCAPE VELOCITY IS "; FNV (M,R)
600 END
```

RUN this program on your Microcomputer and note down the output.

The user can also use function within the function by using a defined function. Here is a program for you to test.

Program 10

```
10 DEF FNS (A,B)= A^2 - 2*A*B+ B^2
20 LET P= FNS (5,4)
30 LET Q= FNS (P,3)
40 LET R= FNS (Q,2)
50 LET X= FNS (FNS(FNS(4,3), 2), 3)
60 PRINT "THE VALUE OF P= FNS (5,4) IS";P
```

```
 70 PRINT " "
 80 PRINT "THE VALUE OF Q= FNS (P,3) IS";Q
 90 PRINT " "
100 PRINT "THE VALUE OF R= FNS (Q,2) IS";R
110 PRINT " "
120 PRINT "THE VALUE OF X= FNS (FNS (FNS (4,3),2)3) IS";X
130 END
```

In this way, as you have seen, a long expression can be used with the help of the DEF statement. Also, a built-in mathematical function can be used within the DEF function. Include the following lines in the above program and RUN.

```
130 LET Y= FNS (SIN (PI/2),5)
140 PRINT "THE VALUE OF Y= FNS (SIN (PI/2),5) IS";Y
150 END
```

Program 11 To calculate the value of a function.

```
10 PRINT "USER DEFINED FUNCTION"
20 PRINT ' "DEF FNA (X,Y,Z)= X^7- Y^6+3*Z^5-3*X*Y*Z +2*
   (X^2)*Y+4*Y^3-8*Z^2"
30 DEF FNA (X,Y,Z)= X^7- Y^6+3*Z^5-3*X*Y*Z+2*
   (X^ 2)*Y+4*Y^3-8*Z^2
40 PRINT ' ' "THE VALUE OF THE FNA (2,4,3) IS"; FNA (2,4,3)
50 END
```

You are advised to evaluate the function given in the above program for the values X = 2, Y = 4 and Z = 3 by using a pen and paper.

Now enter the program in your computer and check your answer.

Program 12

```
10 PRINT "This program handles the strings by using DEF function."
15 PRINT " ": PRINT" "
20 DEF FNT (E$)= "The capacity of a body to do work by virtue of its" + E$+ "
   is called"
30 READ M$,K$,S$,P$
40 PRINT FNT (M$);K$: PRINT " "
50 PRINT FNT (S$);P$: PRINT " "
60 DATA MOTION, KINETIC ENERGY, POSITION, POTENTIAL ENERGY
70 END
```

The output of the above program will be:

```
This program handles the string variables by using the DEF function.
The capacity of the body to do work by virtue of its MOTION is called KINETIC ENERGY.
The capacity of the body to do work by virtue of its POSITION is called POTENTIAL ENERGY.
```

In a similar way, the reader may use the DEF function to prepare some more programs of his/her own choice.

Define Procedure

The facility of defining a procedure is available in the BBC micro. It is similar to the subroutines; however, the expression is slightly different from a subroutine.

As you know, the keyword DEF is used to inform the computer that a 'function' is going to be defined. In a similar manner a user can define a 'procedure' by the key word DEF PROC < name of a procedure >.

Example DEF PROCgame , DEF PROCmenu , DEF PROCa, DEF PROCmean ; etc.

Once the computer has been informed that the procedure exists, it can be called by name anywhere in the program. The computer's control returns in the GOSUB by the keyword RETURN, whereas in procedure the control returns by the statement ENDPROC.

A procedure allows a user to use the local variables which are defined by the statement LOCAL. A local variable will be not available outside that procedure in which it is used. The same variable may be used elsewhere in another part of the program without difficulty. On the other hand, if any variable which is not declared as LOCAL will be available outside the procedure (i.e. in the remaining parts of the program).

Program

```
 10 CLS : PRINT TAB(10,2) "MASTER MIND"
 20 PRINT TAB (5,5) "HELLO! I AM AT YOUR SERVICE."
 30 PRINT TAB (10,7) "OPTION"
 40 PRINT TAB (5,9) "GAME ......1"
 50 PRINT TAB (5,10) "MUSIC ...2"
 60 PRINT TAB (5,11) "QUIZ ......3"
 70 PRINT TAB (5,12) "FINISH ...4"
 80 ' "ENTER YOUR OPTION(1,2,3 or 4)";
 90 INPUT N
100 IF N > 4 THEN SOUND 1, - 10,20,5 : GOTO 80
110 IF N<  0 THEN SOUND 1, - 10,20,5 : GOTO 80
120  IF INT(N)< > N THEN SOUND 1, - 10,20,5 : GOTO 80
130 IF N= 1 THEN PROCgame
140 IF N= 2 THEN PROCmusic
150 IF N= 3 THEN PROCquiz
160 IF N= 4 THEN PRINT "GOOD BYE" : GOTO 2000
170 CLS : PRINT "PRESS RETURN TO CONTINUE"; : INPUT Y $
```

```
180 IF Y $ = " " THEN GOTO 10 ELSE END
190 REM * procedure GAME *
200 DEF PROCgame
210 CLS : PRINT "TOSSING A COIN"
220 LOCAL N
230 INPUT "Number of tosses"; N
240 FOR T = 1 TO N
250 P = RND (2)
260 IF P = 1 THEN PRINT "In toss no. "; T ; ", you got HEAD"
270 IF P = 2 THEN PRINT "In toss no. "; T ; ", you got TAIL"
280 FOR wait = 1 TO 1200 : NEXT wait
290 NEXT T
300 PRINT "Press space bar to continue" : R $ = GET $
310 ENDPROC
480 REM * procedure MUSIC *
490 DEF PROCmusic
495 PRINT "PROCmusic begins"
500 FOR K = 1 TO 20
510 SOUND 1, - 12,240,3 : SOUND 1, - 10,160,3
520 NEXT K
530 VDU 7 : VDU 7 : VDU 7 : VDU 7
540 PRINT "PROCmusic over"
550 ENDPROC
790 REM * procedure QUIZ *
800 DEF PROCquiz
810 LOCAL N, A $
820 FOR N = 1 TO 10
830 READ A $, B $
840 PRINT "WHAT IS THE UNIT OF"; A $
850 INPUT C $
860 IF C $ = B $ THEN PRINT "CORRECT!" ELSE PRINT "WRONG! The
     correct unit is" ; B $
870 FOR W = 1 TO 200 : NEXT W
880 NEXT N
890 DATA CURRENT, AMPERE, POTENTIAL DIFFERENCE, VOLT, WORK,
     JOULE
900 DATA ENERGY, JOULE, ELECTRIC CHARGE, COULOMB, LENGTH,
     METRE
1000 DATA MASS, KILOGRAM, TEMPERATURE, KELVIN, MAGNETIC FIELD,
     TESLA
1010 DATA FORCE, NEWTON
1020 PROCmusic
1030 ENDPROC
```

In this program the computer allows the user to select the procedure by entering 1, 2 or 3. If the user is interested in music, he has to enter the option 2, which shifts the control of the computer to the line number 490 directly. The execution of the line numbers from 490 onwards to 550 will take place. The line number 550 transfers the control of the computer directly to line number 170 of the main program. You have seen the statement on sound and VDU 7 in the above section on PROCmusic. We will discuss these statements in greater detail in the chapter entitled SOUND.

Further, the variable N is used in the procedures game and quiz. At these places we have informed the computer that as this N is our local variable it has no connection with any other value(s) of N in the remaining program by the statement LOCAL.

You will have noticed that in line number 1020 the PROCmusic statement is used. It is to call the procedure music, which is defined in the program. Remember that you can use a procedure in the other procedure, bearing in mind that the procedure music will not consist of the statement PROCquiz, since PROCquiz has PROCmusic in it already.

Enter the above program and RUN it.

Note the change(s) after deleting line number 1020 from the program.

Include the following lines in the program and RUN.

```
205 PROCmusic
295 PROCmusic
```

From the above discussions we conclude that statement DEFPROC is easier to handle than the GOSUB or ON N GOTO statement.

EXERCISES

1. Prepare a program to calculate the time period of a simple pendulum by using the DEF statement for the values L=0.95,1,1.02 and 1.05.
   (Hint DEF FNT(L)=2*(PI)*SQR(L/G)
   where L is the length of the string in metres
   G is the acceleration due to gravity
   (= constant 0.98))
2. By using subscripted data and the DIM statement enter an array of 15 numbers and write a program to find the sum.
3. Point out the mistakes (if any) in the following statements
   (i) DIM (-20)
   (ii) DIM A (20)

```
(iii)  DIM A $ (10)
(iv)   DIM (A=10)
(v)    DIM 100
(vi)   DIM (6,4,2)
(vii)  DIM A $ (4,A)
(viii) DIM A (0,0)
(ix)   DEF FNP (X)=X^2-Y^2
(x)    DEF FN (K)=K*K
(xi)   DEF FNT $ (S $)= "you are" + S $
(xii)  DEF FNX (A,B,C)=(A-B)*(A+B)
(xiii) DEF FN C (R)=2* (PI)*R
(xiv)  DEF FNB9 (X)=45-X*B
(xv)   DEF FNY (A,B)=A^2+2*B*A+B^2
```

4. What will the outputs of the following programs be?

```
(a) 10 DIM A (20)
    20 FOR I=1 TO 20 STEP 2
    30 PRINT A (I)
    40 NEXT I
(b)  10 DIM A $ (4) B (4) C (4) D (4)
     20 REM * Reading the DATA *
     30 FOR N=1 TO 4
     40 READ A $ (N),B (N),C (N),D (N)
     50 NEXT N
     60 REM * PRINTING THE DATA *
     70 FOR N=1 TO 4
     80 PRINT A $ (N),B (N),C (N),D (N)
     90 PRINT " "
    100 DATA ANURAG,28,15,17
    110 DATA MANU, 12,19,28
    120 DATA SUNDARAM,22,18,21
    130 DATA GEETA,25,17,15
    140 END
```

5. The following programs have bug(s). Carefully debug them.

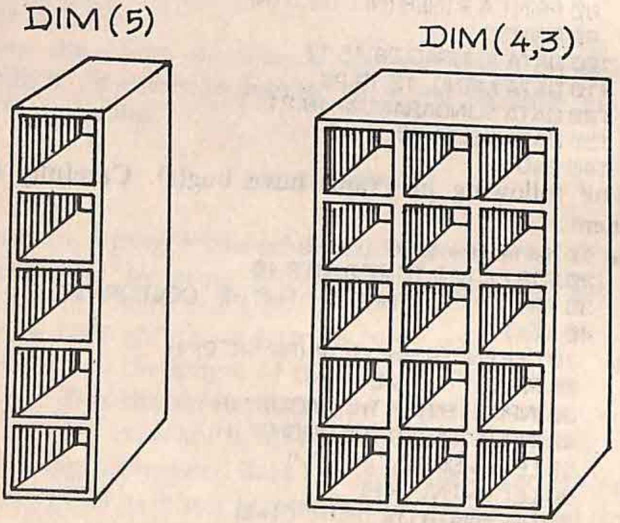
```
(a) 10 def fnc (f)= (5/9)* (F-32)
    20 FOR F=100 TO 200 STEP 10
    30 PRINT F; "FARENHEIT= ";FNC (F); "CENTIGRADE"
    40 NEXT F
(b)  10 DEF FNV (R,H)= (1/3)* (PI)* (R^2)*H
     20 PRINT " C O N E "
     30 INPUT "ENTER THE RADIUS" ;R
     40 INPUT "ENTER THE HEIGHT" ;H
     50 LET L=SQR (h^2+R^2)
     60 LET V=FNV (R,H)
     70 DEF FNA (R,L)= (PI)*R* (R+L)
     80 LET A=FNA (R,L)
     90 PRINT "VOLUME OF THE CONE= ";V
    100 PRINT "TOTAL AREA OF THE CONE= ";A
```

```
110 End
```

6. Explain the following program. Is there any bug in it? If the answer is yes, then remove it.

```
10 CLS
20 PRINT " "
30 LET A $ = "OXFORD UNIVERSITY PRESS"
40 LET B $ = "COMPUTER LITERACY PROJECT"
50 LET C $ = "HAS PUBLISHED A BOOK ON"
60 LET D $ = "N C E R T "
70 LET E $ = "MATHEMATICS"
80 LET F $ = "BASIC PROGRAMMING"
90 DEF FNT (A $,C $,F $) = A $ + C $ + F $
100 PRINT FNT (A $, C $, F $) : PRINT " "
110 PRINT FNT (D $,C $,E $)
120 END
```

7. Write a program by using statement DEFPROC to calculate,(i) the area of a triangle, (ii) the area of a rhombus, (iii) the area of a parallelogram. Present option at the beginning.



1-Dimensional & 2-Dimensional array

CHAPTER 14

GET$ and INKEY$ Statements

When playing a game on the computer, it would be very tedious if you had to keep pressing the RETURN key after each change. So the BBC micro has four very useful BASIC statements which respond to single key strokes. They are GET$, GET, INKEY$ and INKEY.

GET$ and GET Statements

These functions wait until a key is pressed, irrespective of time.

Program 1

```
10 CLS
20 LET A=20: LET B=35
30 PRINT "A+B="; A+B
40 PRINT "PRESS ANY ALPHABET KEY TO FIND A*B"
50 A $=GET $
60 PRINT "A*B="; A*B
70 PRINT "GOOD-BYE"
80 END
```

In line number 50 the computer will wait until the user presses any key; there is no need to press the RETURN key. After the user presses the key, the computer will execute line 60 onwards. If line 50 is replaced by the following:

```
50 A=GET
```

The user has to press any number key or enter a number.

INKEY$ and INKEY Statements

These statements wait for an allotted time. If no key is pressed, it will start executing the next line statement. Replace line number 50 of the above program as follows and RUN the program to see the output:

```
50 A$=INKEY$ (200)
```

Explanation

In the above line 50 the time allotted to press any alphabet key is approximately 2 seconds. If the key is not pressed within 2 seconds the computer starts to execute line number 60.

If a programmer uses INKEY instead of INKEY$ then he/she has to enter a number.

A part of the program in which the computer asks the user for another run is given below for you to try in your program.

Program 2

```
200 PRINT "DO YOU WANT ANOTHER RUN?"
210 A $ = INKEY $ (300) : REM * Time allowed to answer= 3 sec *
220 IF A $ = " " THEN PRINT "LATE!": GOTO 250
230 IF A $ = "Y" THEN PRINT "BRAVE!": GOTO 10
240 IF A $ = THEN PRINT "TIRED!": GOTO 260
250 PRINT "ANSWER AGAIN": GOTO 210
260 PRINT "GOOD BYE"
270 END
```

In line 230 GOTO 10 leads the control to first statement of the main program.

Another program for a similar task is given below.

Program 3

```
200 PRINT "DO YOU WANT ANOTHER RUN?"
210 REPEAT
220 LET K $ = GET $
230 IF K $ = "Y" THEN RUN
240 UNTIL K $ = "N"
250 END
```

If the user does not press Y or N, the computer will be permanently stuck until he or she does so.

Now a program with INKEY$

Program 4

```
 10 CLS
 20 PRINT TAB (3,4) "LET US CHECK HOW FAST YOU ARE"
 30 INPUT "ENTER YOUR FULL NAME (Not smaller than 10 alphabets. ";N $
 40 FOR P= 1 TO 10
 50 X = RND (10)
 60 Q $ = MID $ (N $,X,1)
 70 PRINT TAB (19, 12) "PRESS : ";Q $
 80 Q $ = INKEY $ (1000)
 90 IF INKEY $ (10 00) = Q$ THEN PRINT TAB (3,16) "CORRECT" : GOTO 110
100 PRINT TAB (3,16) "LATE! "; N $
110 FOR W = 1 TO 500 : NEXT W
120 PRINT TAB (3,20) "TRY NEXT"
130 FOR T = 1 to 500 : NEXT T : CLS
140 NEXT P
150 END
```

While running this program the user should be very quick. In line 80 reduce INKEY$(1000) by a lesser number, say 500, to make the game more complicated or vice versa.

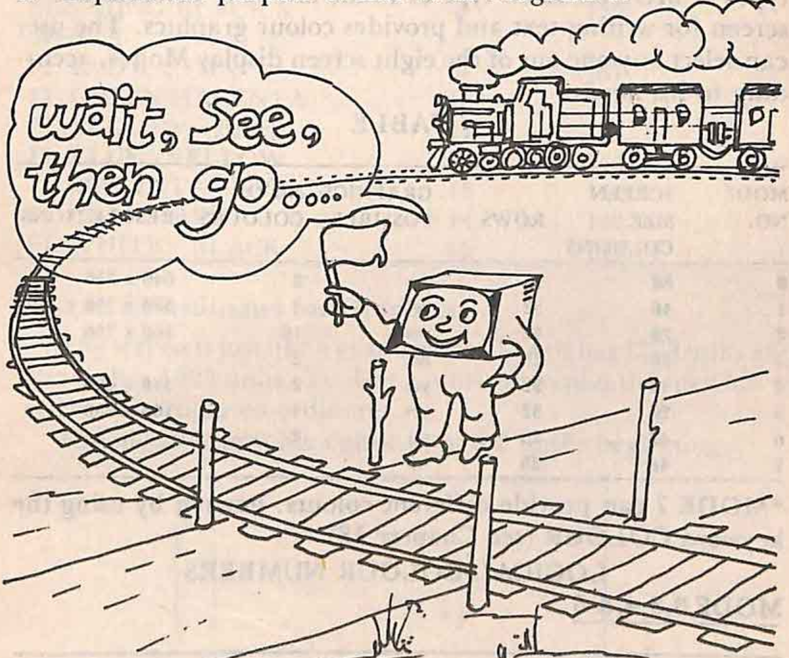
Replace lines 30,40,50,60 as follows

```
30 N $ = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
40 FOR P= 1 TO 26
50 LET X= RND (26)
60 Q $ = MID $ (N $,X,1)
```

RUN this changed program and see the output.

EXERCISES

1. Write the function of the statements GET, GET$. Prepare a PROGRAM using these statements.
2. Explain the meaning of INKEY (800). What will happen if INPUT is used in place of INKEY(800)?
3. What are the differences between the statements given in each of the following pairs:
   (i) GET and GET$ (ii) INKEY and INKEY$?
4. Explain the purpose of line 80 in Program 4.


wait, See, then go....

CHAPTER 15

Advanced Graphics

In the BBC Microcomputer a very good facility that is provided is a set of key words with which the user can write text, draw a figure and produce colour graphics on the screen.

Some main keywords (statements) are COLOUR, GCOL, DRAW, MOVE, PLOT & VDU. Each statement is discussed in brief, one at a time.

MODE Statement

By this statement the user can select a particular graphic MODE. Its format is MODE n.

Here 'n' takes any value from 0 to 7. Thus there are eight types of MODEs. Each type of Mode allows a different size of screen for writing text and provides colour graphics. The user can select any one out of the eight screen display Modes, according to his needs.

TABLE

MODE NO.	SCREEN SIZE COLUMNS	ROWS	GRAPHICS POSSIBLE?	NO OF COLOURS	RESOLUTIONS
0	80	32	yes	2	640 x 256
1	40	32	yes	4	320 x 256
2	20	32	yes	16	160 x 256
3	80	25	no	2	–
4	40	32	yes	2	320 x 256
5	20	32	yes	4	160 x 256
6	40	25	no	2	–
7	40	25	no	*	–

*MODE 7 can provide different colours, but not by using the keyword COLOUR (see Chapter 18)

LOGICAL COLOUR NUMBERS

MODE 0,3,4,& 5

COLOUR	FOREGROUND	BACKGROUND
BLACK	0	128
WHITE	1	129

MODE 1 & 6

COLOUR	FOREGROUND	BACKGROUND
BLACK	0	128
RED	1	129
YELLOW	2	130
WHITE	3	131

MODE 2

COLOUR	FOREGROUND	BACKGROUND
BLACK	0	128
RED	1	129
GREEN	2	130
YELLOW	3	131
BLUE	4	132
MAGENTA	5	133
CYAN	6	134
WHITE	7	135
FLASHING BLACK & WHITE	8	136
FL.RED & CYAN	9	137
FL.GREEN MAGENTA	10	138.
FL.YELLOW – BLUE	11	139
FL.BLUE – YELLOW	12	140
FL.MAGENTA – GREEN	13	141
FL.CYAN – RED	14	142
FL.WHITE – BLACK	15	143

Screen Co-ordinates for Graphics

The screen is just like a graph whose X-axis has 1279 units and Y-axis has 1023 units. To draw, move or to plot,the user has to select the proper co-ordinates.

The invisible graphic cursor is on 0,0 in the beginning.

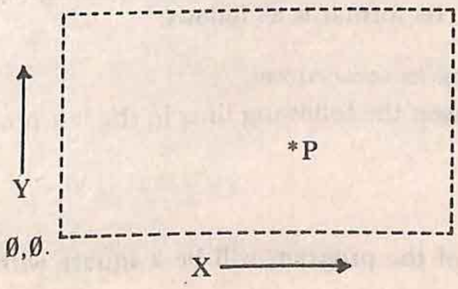


Fig. 15.1

MOVE X,Y Statement

This statement moves the graphics cursor to the indicated co-ordinate without drawing anything on the screen. Its format is as follows:

MOVE X,Y

Program 1 To move the invisible graphics cursor.

```
10 MODE 2
20 MOVE 800,300
```

Explanation

Line number 10 will convert the screen to MODE 2 and line number 20 tells the computer to move the graphic cursor to a point whose co-ordinates are (800,300). It will be approximately at point P in Fig. 15.1 (page 89).

DRAW X,Y Statement

This statement tells the computer to draw a line from the point where the cursor lies whose co-ordinates are (X,Y). Its format is as follows:

DRAW X,Y

Program 2 A program to draw a square.

```
10 MODE 2
20 MOVE 800,300
30 DRAW 800,700
40 DRAW 400,700
50 DRAW 400,300
60 DRAW 800,300
70 END
```

The output of this program will be a square of the length of 400 units, all of whose four sides are in white.

COLOUR n Statement

This statement is used to colour the text, background screen and a text line. Its format is as follows:

COLOUR n

where n stands for colour number.

Program 3 Insert the following lines in the last program:

```
25 GCOL0,1
35 GCOL0,2
45 GCOL0,3
55 GCOL0,4
```

The output of the program will be a square with different coloured sides.

Program 4 A program to give a colour effect to the text.

```
10 MODE 2
20 FOR C= 1 TO 7
25 COLOUR C
30 PRINT TAB (5) "COLOUR EFFECT";C : PRINT " "
40 NEXT C
50 END
```

Enter the program and see the output yourself. Also remove line 25 and see the output. Note the difference between these two programs.

Program 5 A program which colours the screen

```
10 MODE 1
20 COLOUR 130: CLS : REM *COLOUR THE SCREEN*
30 COLOUR 0
40 PRINT TAB (4,8) "JAWAHARLAL NEHRU"
50 END
```

GCOL0, n Statement

This statement stands for Graphic COLour. This is used to shade in colour lines and areas. GCOL is followed by its code number 0 and then its colour number denoted by n. Its format is as follows,

GCOL0, n

where n stands for colour number. It should be selected in accordance with your working mode and choice.

In MODE 5 these following statements are used for foreground colour:

GCOL0, 0 for black colour
GCOL0,1 for red
GCOL0,2 for yellow
GCOL0,3 for white

Whereas the following are used for background colours:

GCOL0,128 for black
GCOL0,129 for red
GCOL0,130 for yellow
GCOL0,131 for white

Program 5 A program to print three red stars on a blue background.

```
10 MODE2: COLOUR 132 : CLS
20 COLOUR 2
30 PRINT TAB (9,10) "*";"*";"*"
40 END
```

PLOT 85, X, Y Statement

This statement draws and fills in a triangle drawn from the last two plotted points to the point defined by (X,Y) coordinates. The colour is the latest foreground colour used in the program.

Program 6 A program to draw a coloured triangle.

```
10 MODE 2: COLOUR 133: CLS
20 MOVE 200, 200
30 GCOL0, 3
40 DRAW 100, 200
50 PLOT 85, 600, 600
60 END
```

Explanation

Line number	Function
10	Puts computer in MODE 2 and colours the whole screen magenta.
20	Moves the graphic cursor to a point whose coordinates are 200, 200
30	Selects the foreground graphic colour as green.
40	Draws a line from the point 200, 200 to the point, 1000, 200
50	Joins the above three points and colours it in green which is the latest colour assigned in line 30.
60	Ends the program.

Program 7 A program to draw the 'Red Cross' symbol

```
10 MODE 2 : COLOUR 132 : CLS
20 MOVE 500, 100 : GCOL0, 1: REM* colour*
30 DRAW 700, 100
40 PLOT 85, 700, 900
50 RAW 500, 900
60 PLOT 85, 500, 100
70 MOVE 200, 400
80 DRAW 200, 600
90 PLOT 85, 1000, 400
100 DRAW 100, 600
110 PLOT 85, 200, 600
```

PLOT 69,X,Y Statement

This statement marks the point defined by X,Y coordinates.

Program 8 A program which puts a mark at a point whose co-ordinates are 600, 400 is given below:

```
10 MODE 5
20 COLOUR 129: CLS
30 PLOT 69, 600, 400
40 END
```

Program 9 To draw a line Y = X/2.

```
10 MODE 2 : COLOUR 133 : CLS
20 FOR X= 20 TO 1200
30 LET Y=X/2
40 PLOT 69,X,Y : REM * marks the point *
50 NEXT X
60 PRINT TAB(5,3) "GRAPH Y=X/2"
70 FOR N= 20 TO 1200
80 PLOT 69,N,20
90 NEXT N
100 FOR M= 20 TO 1000
110 PLOT 69, 40,M
120 NEXT M
130 END
```

EXERCISES

1. State the colour corresponding to the following numbers in MODE 2: COLOUR 132, COLOUR 10, COLOUR 4, COLOUR 140, COLOUR 15.
2. Write down the colour code corresponding to the following colours in MODE 5: black foreground, white background.
3. What is the size of the text screen in following MODEs? MODE 2, MODE 7, MODE 5.
4. Write down the formats of the following statements: DRAW, MOVE, PLOT, COLOUR and MODE.
5. Prepare a program to draw a boat against a blue background.
6. Prepare a program to draw the graph of Y = (3/2)*X + 5 by using PLOT 69,X,Y statement.
7. Point out the errors, if any, in the following statements:

```
(a) 10 MODE 8
(b) 20 move 600;500
(c)140 DRAW (800,350)
(d)120 PLOT 69,1000,1280
(e)150 MOVE 0,560,434
(f) 60 COLOUR 25
(g)110 DRAW -369, -500
```

```
(h)180 mode 5 : COLOUR 6
```

8. Debug the following program, if necessary, and RUN:

```
(a) 10 MODE 3
    20 COLOUR 142 : CLS : REM Background colour
    30 GCOL0,2
    40 MOVE 300,300
    50 DRAW 600,300
    60 PLOT 85,600,800
    70 COLOUR 4
    80 PRINTTAB (5,2) ''RIGHT TRIANGLE''
    90 END
(b) 10 MODE 2
    20 COLOUR - 132 : CLS
    30 MOVE 0,0
    40 REM * TO DRAW SQUARE *
    50 DRAW 500,0
    60 GCOL0,3
    70 PLOT 85,500,300
    80 DRAW 0,300
    90 PLOT 0,0
   100 PRINTTAB (5,2) ''square''
   110 END
(c) 10 REM * to draw box *
    20 MODE 2 : colour 129 : CLS
    30 FOR X=0 TO 1000 STEP 20
    40 DRAW 1000-X,X
    50 DRAW 1000-X,1000-X
    60 DRAW X, 1000-X
    70 DRAW X,X
    80 next X
    90 END
```

9. Try the following programs:

```
(a) 10 MODE 2
    20 X=200 : Y=300
    30 D=500
    40 MOVE X,Y
    45 GCOL0,1
    50 DRAW X,Y+D
    55 GCOL0,2
    60 DRAW X+D,Y+D
    65 GCOL0,3
    70 DRAW X+D,Y
    75 GCOL0,4
    80 DRAW X,Y
    90 END
```

(b)
```
10 MODE 2
20 FOR CX=200 TO 900 STEP 100 : CY=500
30 R=50
40 FOR X= -R TO R : Y=SQR (R*R-X*X)
50 PLOT 69, CX+X, CY+Y
60 PLOT 69, CX+X, CX-Y
70 NEXT X
80 NEXT CX
```

Note the output yourself. Modify the program, if there is any bug.

10. Enter and RUN the following program on your Microcomputer.

```
 10 MODE 4
 20 WAIT=INKEY (100)
 30 FOR P=1 TO 160 + RND (160)
 40 PLOT 69, RND (1279), RND (1023)
 50 NEXT P
 60 WAIT=INKEY (100)
 70 PRINT TAB (0,3); "FULL MOON"
 80 REM CENTRE 640, 512 and RADIUS 450
 90 CX=640 : CY=512 : R=450 : A= -5
100 MOVE CX,CY+R
110 PLOT0,0,0
120 FOR Y=R TO-R STEP A
130 X=SQR (R*R-Y*Y)
140 PLOT 85, CX-X, CY+Y
150 PLOT 85, CX+X, CY+Y
160 NEXT Y
170 END
```

(i) Delete line 110 and RUN the program.

(ii) The user is also advised to change the value of R=150 in line 90. RUN the modified program and note down the output.

CHAPTER 16

Making a Character

If a user wants to write a program for a game in which aeroplanes or cars move about the screen, he/she needs some characters which are different from those on the key. For this task the BBC Microcomputer has some spare code numbers and the user may design his/her own characters in accordance with particular requirements. The AISCII code numbers available for these purposes are 224 to 255 in MODEs 0 to 6 but not in MODE 7.

I want to print these CHR$

Fig. 16.1

For making the required character the user needs eight by eight graph paper. The user can draw characters by shading in some of the squares in accordance with requirements of the figure.

Here is one such designed character, a car (see Fig. 16.2),

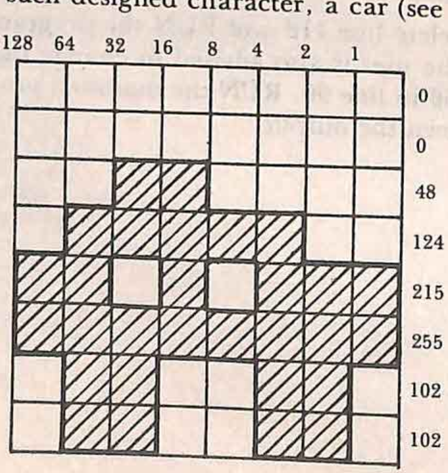


Fig. 16.2

To assign suitable values to the columns, these are numbered as above.

Now look at row 1 and see which of its squares is shaded in and add the number in each row.

Row 1 is blank so the sum of the shaded squares = 0

Row 2 is blank so the sum of the shaded squares = 0

Row 3, Two squares are shaded so the sum of their column numbers is (32 + 16) = 48

Row 4, Five squares (64 + 32 + 16 + 8 + 4) are shaded so that their sum is 124.

Similarly, the sum of the shaded columns in Row 5, Row 6, Row 7, Row 8 are 215,255,102 & 102, respectively.

These eight numbers instruct the computer to store the character in its memory for which we may assign any number from 224 to 255; let us assign number 230.

The complete instruction for making this character will be:
VDU 23,230,0,0,48,124,215,255,102,102

Program 1 To print a CAR

```
10 MODE 2 : COLOUR 132 : CLS
20 COLOUR 4
30 VDU 23,230,0,0,48,124,215,255,102,102
40 LET CAR $ = CHR $ (230)
50 PRINT TAB (10,8) CAR $
60 END
```

In line 30 VDU 23 is a statement for making a character, 230 is the character code, the other eight numbers after 230 are the sum of the squares in row 1 to row 8 (separated by commas).

In line 40 the computer assigns CHR $ (230) as CAR $.

In line 50 the computer prints the car.

Program 2 To move a CAR

The user is advised to add the following lines in PROGRAM 1:

```
45 FOR C= 1 TO 18 : REM * Column 1 to 18 *
50 PRINT TAB (C,8) " " + CAR $: REM * prints car in column c & 8th row *
60 FOR W= 1 TO 300 : NEXT W : REM * waiting time *
70 NEXT C
80 END
```

Remove line 60 from the above program and note the effect.

How Are Characters Added?

Suppose you wish to draw a truck. For this two extra characters should be addded because of its length.

In the following program lines 30 and 40 inform the computer about these two characters which are the two parts of the truck. In line 50 the computer adds the two characters.

The front part of the truck is given in Fig. 16.3.

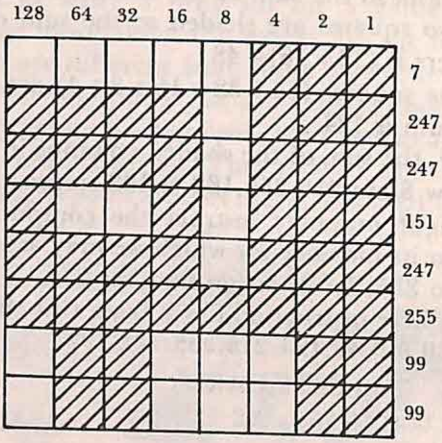


Fig. 16.3

Let its code number be 225
VDU 23,225,7,247,247,151,247,255,99,99

The back part of the truck has the code number 226. For this the complete computer instruction will be as follows:
VDU 23,226,255,255,255,255,255,255,6,6

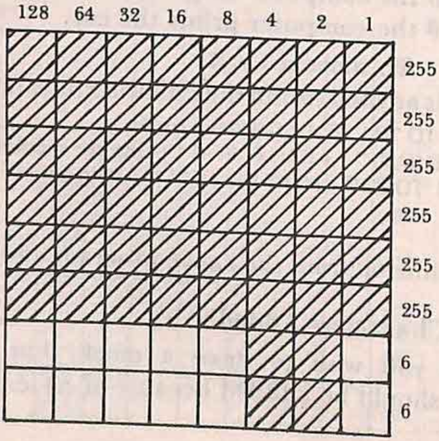


Fig. 16.4

Program 3 A moving truck

```
 10 MODE 2 : COLOUR 132 : CLS
 20 COLOUR 2
 30 VDU 23,225,7,247,247,151,247,155,99,99
 40 VDU 23,226,255,255,255,255,255,255,6,6
 50 LET TRUCK $ = CHR $ (225) + CHR $ (226)
 60 FOR C= 18 TO 1 STEP- 1
 70 PRINT TAB (C,10) TRUCK $; " "
 80 FOR W= 1 TO 600 : NEXT W
 90 NEXT C
100 END
```

In line 50 the two characters are added and in line 70 they are printed. The blank space after the truck erases the truck. In line 80 it allows waiting time. The user is advised to delete line 80 and see what happens.

Program 4 A caterpillar

```
 10 MODE 2 : COLOUR 133 : CLS
 20 VDU 23,239,6,63,127,223,255,127,63,10
 30 VDU 23,231,85,255,255,255,255,255,255,170
 40 VDU 23,232,84,252,255,255,254,252,248,160
 50 LET CATER $ = CHR $ (230) + CHR $ (231) + CHR $
    (231) + CHR $ (232)
 60 FOR C= 14 TO 1 STEP - 1
 70 COLOUR 3
 80 PRINTTAB (C,10) CATER $ + " "
 90 FOR W= 1 TO 1000: NEXT W
100 NEXT C
110 END
```

Here is a program with special characters for you to try on your Micro. The user is advised to draw the character on the graph.

Program 5

```
 10 MODE 2 : COLOUR 134 : CLS
 20 VDU 23,230,8,28,28,107,127,107,8,28
 30 VDU 23,231,8,28,62,127,62,28,8,0
 40 VDU 23,232,54,127,127,62,28,8,0
 50 VDU 23,233,8,28,62,127,127,127,28,62
 60 LET CLUB $ = CHR $ (230)
 70 LET DI $ = CHR $ (231)
 80 LET HEART $ = CHR $ (232)
 90 LET SPADE $ = CHR $ (233)
110 PRINT " " : PRINT " "
120 LET N= RND (4)
130 IF N= 1 THEN COLOUR 0 : PRINT CLUB $;" "
140 IF N= 2 THEN COLOUR 1 : PRINT CLUB D1 $; " "
150 IF N= 3 THEN COLOUR 1 : PRINT HEART $ ; " "
```

```
160 IF N=4 THEN COLOUR 0 : PRINT SPADE $ ; " "
170 PRINT " " : PRINT " " : PRINT " "
180 PRINT "TYPE RUN TO RESTART & PRESS THE RETURN KEY"
190 END
```

How to Join Characters Vertically

Suppose you need to draw a rocket. To do this you have to join the characters vertically. First you have to form the characters on a graph. Note down the characters for the rocket as follows:

```
VDU 23,240,16,16,16,56,124,124,124,124
VDU 23,241,214,124,124,124,124,124,124,124
VDU 23,242,124,124,124,254,254,40,40,40
```

The user is advised to draw the graph for these characters and see the figure for himself.

Program 6 To launch a ROCKET

```
 10 MODE 2: COLOUR 130 : CLS
 20 VDU 23,240,16,16,16,56,124,124,124,124
 30 VDU 23,241,214,124,124,124,124,124,124,124
 40 VDU 23,242,124,124,124,254,254,40,40,40
 50 FOR R= 25 TO 1 STEP - 1
 60 COLOUR 4
 65 PRINT TAB (10, R+2) " "
 68 PRINT TAB (10, R+3) " "
 70 PRINT TAB (10,R) CHR $ (240)
 80 PRINT TAB (10,R+ 1) CHR $ (241)
 90 PRINT TAB (10,R+ 2) CHR $ (242)
100 FOR W= 1 TO 500 : NEXT W
120 NEXT R
130 END
```

Remove the line 100 of PROGRAM 6 and see the effect on the output.

You can introduce a launching sound when you learn about the BBC computer's SOUND effects. For the time-being just insert the following lines in the above program.

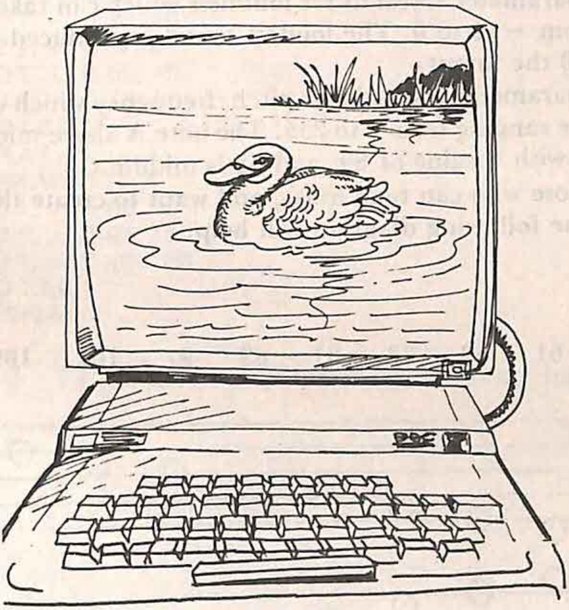
```
55 SOUND 0, - 15,20,100
56 SOUND 0, - 7,20,100
```

RUN this modified program.

EXERCISES

1. What is the use of the VDU23 statement? Can you prepare a special character which is not in the keyboard without a VDU23 statement?

2. Draw the graph for the following VDU statements and try to PRINT these on your Microcomputer screen.
   - (i) VDU 23,230,8,28,28,107,127,107,8,28
   - (ii) VDU 23,231,265,20,20,116,84,116,4,4
3. Prepare a helicopter and launch it. (Introduce sound effect when you learn how to do it.)
4. Prepare a character for a doll and move it across the screen.
5. How many special characters can you design in a program?



CHAPTER 17

Sound

The BBC Microcomputer has four channels to produce sounds. A user can play music or produce sound with a program. The basic key word which causes the computer to produce sound is SOUND. The key word SOUND produces a range of sound effects specified by the numbers or variables following the command. Its format is as follows:

```
SOUND C,L,P,D
```

The parameter C stands for channel number which can take any value, viz. 0,1,2 and 3. Channel number 0 causes noise whereas channels 1,2 and 3 produce musical sounds.

The parameter L stands for loudness which can take any integer from − 15 to 0. The loudest sound is produced on − 15 and on 0 the lowest.

The parameter P stands for pitch/frequency which can take any value ranging from 0 to 255. The note A above middle C is selected with a value of 89, and 53 is middle C.

For those who can read music and want to create their own tunes, the following diagram will help.

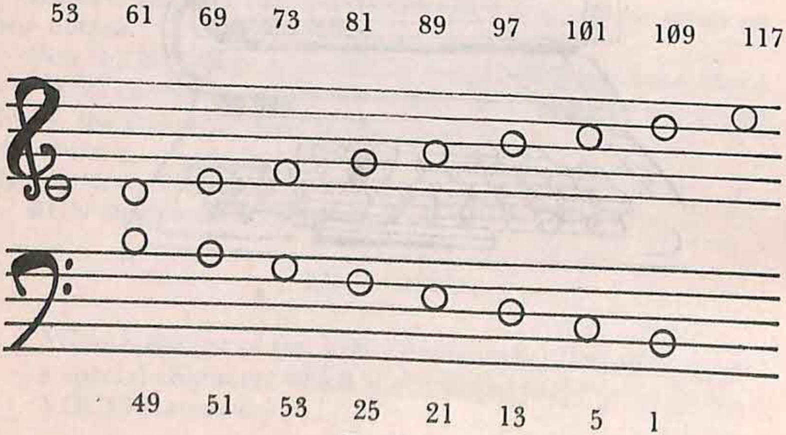


Fig. 17.1

The parameter D stands for duration which can take any value from − 1 to 254. This value is the range 0 to 254. It gives a note duration of that number multiplied by one-twentieth of a second. Thus if D = 50 the note will last for 2.5 seconds. If D = − 1 then the note will continue to produce the sound until you press the ESCAPE key to interrupt the PROGRAM.

Let us try these sound programs.

Program 1

```
10 CLS : PRINT TAB (10,12) "SOUND"
20 SOUND 1, - 12, 97, 10
30 SOUND 1, - 12, 105, 10
40 SOUND 1, - 10, 89, 20
50 SOUND 1, - 15, 41, 15
60 SOUND 1, - 5, 69, 20
70 END
```

Program 2

```
 10 FOR PLAY = 1 TO 2
 20 SOUND 1, - 10, 73, 10
 30 SOUND 1, - 10, 81, 01
 40 SOUND 1, - 10, 89, 10
 50 SOUND 1, - 10, 73, 10
 60 SOUND 1,0,0,1
 70 NEXT PLAY
 80 FOR REPLAY = 1 TO 2
 90 SOUND 1, - 10, 89, 10
100 SOUND 1, - 10, 93, 10
110 SOUND 1, - 10, 101, 20
120 SOUND 1,0,0,1
130 NEXT REPLAY
140 END
```

Program 3 To produce sound effect on different channels.

```
10 FOR C = 0 TO 3 STEP 1
20 CLS : PRINT TAB (3,8) "SOUND EFFECT IN CHANNEL NUMBER ";C
30 SOUND C, - 15,89,20
40 INPUT TAB (3,15) "PRESS SPACE BAR TO CONTINUE"; K $
50 NEXT C
60 END
```

Program 4 To produce sound effects of different pitches.

```
10 FOR PITCH = 100 TO 200 STEP 20
20 CLS : PRINT TAB (8,8) "SOUND EFFECT ON PITCH"; PITCH
30 SOUND 1, - 12,PITCH, 20
40 INPUT TAB (3,15) "PRESS ANY NUMBERED KEY";K
50 NEXT PITCH
60 END
```

Program 5 The sound of a MOTORCYCLE

```
10 CLS : PRINT TAB(12,12) "MOTORCYCLE"
20 SOUND 0, - 15,3,100
30 FOR P= 100 TO 200
40 SOUND 1, - 2,P,1
50 NEXT P
60 END
```

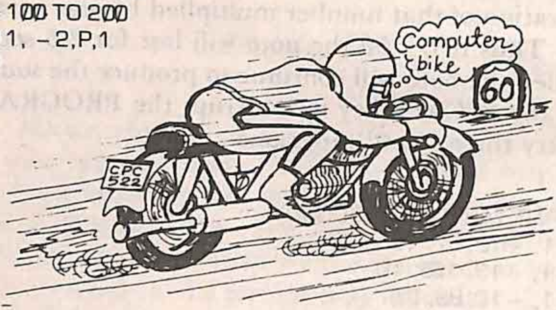


Fig. 17.2

Program 6 A musical tune.

```
10 CLS : PRINT TAB(10,8)"** TUNE**"
20 FOR PLAY = 1 TO 2
30 SOUND 2, - 15,49,20
40 SOUND 1. - 15,61,20
50 SOUND 2. - 15,41,20
60 SOUND 1, - 15,77,20
70 SOUND 2, - 15,33,40
80 SOUND 1, - 15,81,40
90 SOUND 2, - 15,49,40
100 SOUND 1, - 15,81,10
110 SOUND 1, - 15,77,10
120 SOUND 1, - 15,81,10
130 SOUND 1, - 15,89,30
140 NEXT PLAY
150 END
```

Hindustani Classical Music

You can compose Hindustani classical music on your BBC Microcomputer with the help of the following pitches. To prepare your own tune, you should know the fundamentals of HINDUSTANI music. Approximate pitch/frequency numbers are given in the brackets corresponding to the Hindustani swara, for your BBC Microcomputer.

Sa(56), Re(64), Ga(72), Ma(76), Pa(84), Dha(92), Ni(100)
Higher Sa(104)

Try the following program.

Program 7 Sargam of Hindustani music.

```
10 FOR N= 1 TO 5
20 FOR P= 1 TO 8
30 READ A
```

```
 40 SOUND 1, - 15,A,30
 50 NEXT P
 60 FOR W= 1 TO 250 : NEXT W
 70 FOR R= 1 TO 8
 80 READ B
 90 SOUND 3, - 15,B,30
100 NEXT R
110 FOR W= 1 TO 250 : NEXT W
120 RESTORE
130 NEXT N
140 DATA 56,64,72,76,84,92,104
150 DATA 104,100,92,84,76,72,64,56
```

In the above program the numeric variables N,P and R are used to repeat the steps.

You can prepare the Hindustani classical Raag 'LALIT' by using SOUND statements with pitches/frequency of the following Aaroha and Avaroha.

Aaroha :	Ni lower tivra(129), Komal Re(137), Tivra Ga(149), Tivra Ma(157), Komal Ma(153), Tivra Ni(177), Higher Sa(181)
Avaroha :	Higher Sa(181), Tivra Ni(177), Komal Dha(165), Tivra Ma(157), Komal Ma(153), Tivra Ma(157), Tivra Ga(149), Komal Re(137), Sa(133).

Here is a program for you to try on your BBC Micro.

Program 8 A HINDUSTANI RAAG 'SARANG'.

```
 10 MODE 2 : COLOUR 132 : CLS : COLOUR 2
 30 PRINT TAB (8,6) "SARANG" : COLOUR 3
 40 PRINT TAB (8,7) "= = = = = = ="
 50 FOR S= 1 TO 2
 60 FOR P= 1 TO 10
 70 READ F
 80 SOUND 1, - 14,F,7
 90 NEXT P
100 RESTORE
110 NEXT S
120 DATA 133,141,153,161,177,181
130 DATA 181,177,161,153,141,133
140 FOR X= 1 TO 3
150 FOR Q= 1 TO 10
160 READ G
170 SOUND 1, - 15,G,10
180 NEXT Q
190 RESTORE
200 NEXT X
```

```
210 RESTORE
220 FOR R= 1 TO 10
230 READ H
240 SOUND 3, - 10,H,10
250 NEXT R
260 RESTORE
270 FOR S= 1 TO 10
280 READ M
290 SOUND 1, - 14,M,15
300 NEXT S
310 RESTORE
320 END
```

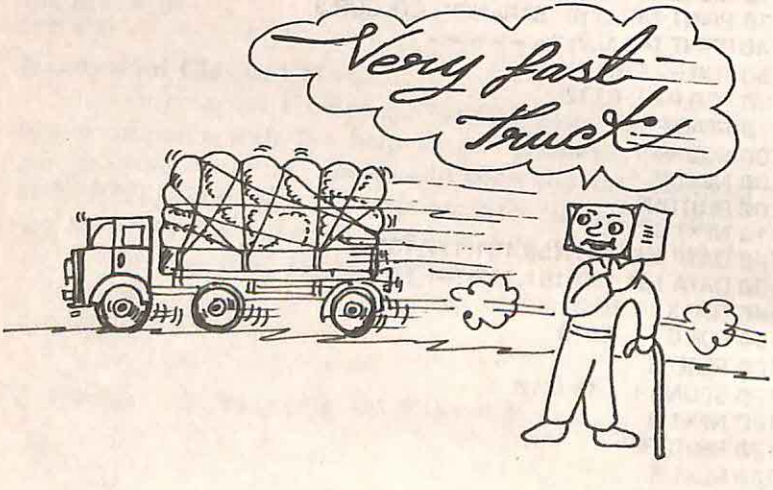
In the above program the pitch/frequency of the Aaroha and Avaroha are given below.

Aaroha	:	Sa(133), Tivra Ma(141), Komal Ma(153), Pa(161), Tivra Ni(177), Higher Sa(181)
Avaroha	:	Higher Sa(181), Tivra Ni(177), Pa(161), Komal Ma(153), Tivra Ma(141), Sa(133)

A knowledge of music would help you to produce a better combination of these swaras.

EXERCISES

1. What are the different parameters used with the SOUND statement? Also write down the ranges of these parameters.
2. Prepare a program of a moving car and use SOUND statement to add sound.



3. Use the following Aaroha and Avaroha to prepare the Hindustani Raag MALKAUNS.
   Aaroha : Sa(133), Ga(145) Ma(153) Dha(165) Ni(173) Sa(181)
   Avaroha : Sa(181) Ni(173) Dha(165) Ma(153) Ga(145) Sa(133)
4. Write down the pitch numbers corresponding to the following Hindustani swara.
   Sa, Tivra Ma, Komal Re, Komal Dha, Tivra Ni, Higher Sa and Pa.
5. Point out the errors in the following expressions:

```
(a) 20 SOUND 12,-15,6,20
(b) 50 SOUND 1,10,50,15
(c) 21 SOUND 3.-7,500,25
(d) 25 SOUND 2,-10,151,-20
(e) 65 SOUND 3,2,-6,125,5
(f) 80 SOUND 0,-12,56,3,50
(g) 45 SOUND 0,0,-8,181,10
(h) 60 SOUND 2,-15,200,20
[i] 5R SOUND 3,-10,101,25
[j] 42 SOUND 2,-10,83,20.
```

CHAPTER 18

Teletext Control

In MODE 7, the BBC Micro provides a direct presentation of the text on the screen. Sometimes it is essential to print in different colours on the screen to attract the attention of the user to specific points. For this purpose the TELETEXT CONTROL is very useful.

You know that in MODE 7 the display consists of 25 rows and 40 columns. Thus, in this MODE, there is a good alphanumeric structure and more space available on the screen.

Here are the following BASIC statements and keys which play an important role in COLOUR display in MODE 7.

1. CHR$ statements
2. VDU statements
3. USER-DEFINED KEYS

CHR$ Statements

These are used with the code numbers assigned to the different colours in MODE 7.

Here is a list of the CHR$ colour control code for Alphanumeric and Flashing effect.

CHR$ code no.	Colour/Effect
129	RED
130	GREEN
131	YELLOW
132	BLUE
133	MAGENTA
134	CYAN
135	WHITE
136*	Flashing effect on
137*	Flashing effect off

*Code nos. 136 and 137 can also be used with the colour code no(s) given in the table.

Here is a program for you to try.

Program 1 To display the colour effect in MODE 7.

```
10 MODE 7
20 PRINT CHR $ (129); "THIS IS A RED FORT"
30 PRINT CHR $ (131); "THIS IS A YELLOW SUNFLOWER"
40 PRINT CHR $ (132); "THIS IS A BLUE SKY"
50 PRINT CHR $ (130): "THIS IS A GREEN TREE"
60 PRINT CHR $ (136); "THIS IS A MAGENTA FLASHING MAGIC"
70 PRINT CHR $ (136); CHR $ (129);CHR $ (157);CHR $ (132); "BLUE
   FOX ON THE RED ROAD"
80 REM * CHR $ (157) for a red background colour CHR $ (129) *
```

Explanation of line 70:

CHR$ (136) for flashing effect; CHR$ (129) for red colour; CHR$ (157) for background and CHR$ (132) for blue text.

SOME MORE CHR$ CODES

CHR$ code no.	Purpose
137	To turn off the flashing in the line after some characters.
141	To produce double-height characters.
140	To turn off the double-height in that line after some characters in double-height.
157	To produce a new background colour.

Program 2

```
10 REM * TO PRODUCE A DOUBLE-HEIGHT ALPHANUMERAL *
20 MODE 7
30 PRINT CHR $ (141); CHR $ (129); "B O M B A Y"
40 PRINT CHR $ (141); CHR $ (129); "B O M B A Y"
50 PRINT " "
60 PRINT CHR $ (141); CHR $ (131); CHR $ (157); CHR $ (132); CHR $
   (136); "D O-O R D A R S H A N"
70 PRINT CHR $ (141); CHR $ (131); CHR $ (157); CHR $ (132); CHR $
   (136); "D O O R D A R S H A N"
80 PRINT " "
90 PRINT CHR $ (141); CHR $ (130); CHR $ (157); CHR $ (136); CHR $;
   (133); "P R E S E N T S"
110 PRINT CHR $ (141); CHR $ (130); CHR $ (157); CHR $ (136); CHR $
   (133); "P R E S E N T S".
120 PRINT CHR $ (141); CHR $ (132); CHR $ (157); CHR $ (136); CHR $
   (129); "C H I T R A H A R"
130 PRINT CHR $ (141); CHR $ (132); CHR $ (175); CHR $ (136), CHR $
   (129); "C H I T R A H A R"
140 PRINT " "
150 END
```

The output of the above program will have double-height colour flashing characters on a coloured background.

```
BOMBAY
DOORDARSHAN
PRESENTS
CHITRAHAR
```

Enter this program and RUN on your BBC Micro.

The VDU Statement

The VDU statement is also used to print alphanumerical characters and to get colour in MODE 7. The following is a sample program. For details see the USER GUIDE.

Program 3

```
 10 CLS
 20 MODE 7
 40 VDU 136 : REM FLASHING
 50 VDU 134 : REM CYAN COLOUR
 60 REM O
 70 VDU 207 : FOR W=1 TO 2000 : NEXT
 80 REM X
 90 VDU 216 : FOR W=1 TO 2000 : NEXT
100 REM F
110 VDU 70 : FOR W=1 TO 2000 : NEXT
120 REM O
130 VDU 207 : FOR W=1 TO 2000 : NEXT
140 REM R
150 VDU 210 : FOR W=1 TO 2000 : NEXT
160 REM D
170 VDU 196 : FOR W=1 TO 2000 : NEXT
180 PRINT " "; : REM * SPACE *
190 VDU 136 : REM FLASHING
200 VDU 129 : REM RED COLOUR
210 REM P
220 VDU 80 : FOR W=1 TO 2000 : NEXT
230 REM R
240 VDU 210 : FOR W=1 TO 2000 : NEXT
250 REM E
260 VDU 197 : FOR W=1 TO 2000 : NEXT
270 REM S
280 VDU 211 : FOR W=1 TO 2000 : NEXT
290 REM S
300 VDU 211 : FOR W=1 TO 2000 : NEXT
310 PRINT " ": PRINT " " : PRINT " "
320 GOTO 40
330 END
```

User-Defined Keys for Colour Effect in MODE 7

Typing in CHR$ followed by its code number within brackets every time is very tedious. Another easier facility is given in the BBC Micro to show the colour effects in MODE 7 by using the USER-DEFINED KEYS in combination with the SHIFT key to obtain special colour effects.

SHIFT & USER DEFINED KEY No.	COLOUR/EFFECT
f0	Does nothing
f1	Red
f2	Green
f3	Yellow
f4	Blue
f5	Magenta
f6	Cyan
f7	White
f8	Flashing on
f9	Flashing off

While using the User-Defined Keys, you will see the colour in the LIST of your program as well as during the execution of the program.

If you want to decorate your program use these Defined Keys.

In the following program you will see some special symbols just after the quotation marks in each of the PRINT statement. They are there because of the SHIFT key and the user defined key number indicated against them. The user has to hold down the SHIFT key and then press the user defined key (red keys) number indicated in the statement. For example,

Symbol ▬ is there in line 10 due to SHIFT key and the user defined key F1 to give red colour. The user has to act as per instructions given in PRINT statement in the brackets. He/She need not type the matter given in the brackets.

Program 4

```
10 PRINT "▬(Press SHIFT & F1 keys together) for red colour"
20 PRINT "▬(Press SHIFT & F3 keys together) for yellow colour"
30 PRINT "▬(Press SHIFT & F6 keys together) for magenta colour"
40 PRINT "■(Press SHIFT & F8 keys together) for flashing ON."
```

On the screen you will not see these symbols in the LIST of your program but you will see them when you print the LIST on paper using the printer.

WINDOW

Sometimes it is very essential to write and draw separately on the screen. Initially it is possible to write the text or to draw anywhere on the screen,but after setting a window the user can write or draw in the specified area for the purpose on the screen.

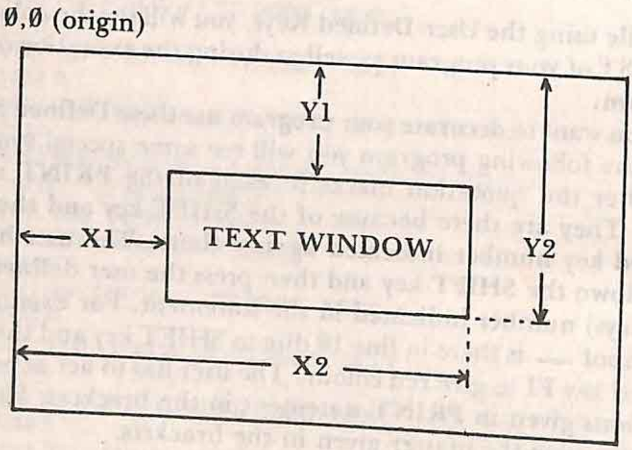
There are two types of windows:

(a) THE TEXT WINDOW
(b) THE GRAPHICS WINDOW

The Text Window It is there to specify the area where the user can write only the text. To form a text window the user has to use the code VDU 28, the format of which is:

VDU 28, X1, Y1, X2, Y2.

Here is a figure for you to study carefully.



The value of the parameters (X1, Y1) and (X2 and Y2) depend on the MODE. The value of X and Y stand for columns and rows. Refer to the chapter on Advanced Graphics for the value of columns and rows in different MODE(s). The origin of the text cursor is initially at the top left.

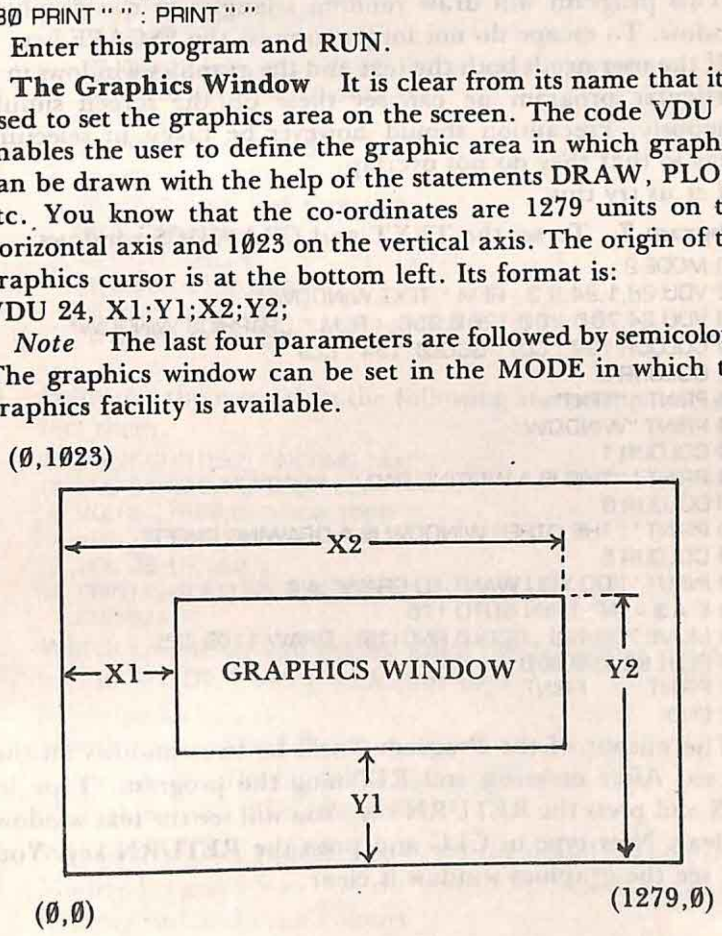
Program 5

```
 10 MODE 1
 20 REM * TEXT WINDOW *
 30 VDU 28,5,20,30,10
 40 COLOUR 129 : CLS : REM * background colour *
 50 COLOUR 2 : REM * foreground colour *
 60 PRINT TAB(6)"  **************** "
 70 PRINT TAB(6) " ****            **** "
 80 PRINT TAB(6) "*                  * "
 90 PRINT TAB (6) "      I N D I A      "
100 PRINT TAB (6) "*                  * "
110 PRINT TAB (6) "****            **** "
120 PRINT TAB (6) "****************** "
130 PRINT " " : PRINT " "
```

Enter this program and RUN.

The Graphics Window It is clear from its name that it is used to set the graphics area on the screen. The code VDU 24 enables the user to define the graphic area in which graphics can be drawn with the help of the statements DRAW, PLOT, etc. You know that the co-ordinates are 1279 units on the horizontal axis and 1023 on the vertical axis. The origin of the graphics cursor is at the bottom left. Its format is:
VDU 24, X1;Y1;X2;Y2;

Note The last four parameters are followed by semicolons. The graphics window can be set in the MODE in which the graphics facility is available.



Let us type in the following program to set the graphics window.

```
 10 MODE 2 : COLOUR 129 : CLS : REM * red background *
 20 REM * graphic window *
 30 VDU 24, 150;300;110;750;
 40 REM * random triangles *
 50 MOVE 150, 300
 60 X= RND (1000) : Y= RND (800)
 70 DRAW X,Y
 80 PLOT 85, X+ 200, Y- 100
 90 GCOL 0, RND (15)
100 GOTO 75
```

This program will draw random triangles in the graphics window. To escape do not forget to press the ESCAPE key.

If the user needs both the text and the graphics windows in a particular program he can set these on the screen simultaneously. Precaution should however be taken in selecting areas so that they do not overlap.

Let us try this:

Program 7 To set the TEXT and GRAPHICS windows.

```
 10 MODE 2
 20 VDU 28,1,24,9,3 : REM * TEXT WINDOW *
 30 VDU 24,700;200;1200;950; : REM * GRAPHICS WINDOW*
 40 COLOUR 132 : CLS : GCOL0, 134 : CLS
 50 COLOUR 2
 60 PRINT ' "TEXT"
 70 PRINT "WINDOW"
 80 COLOUR 1
 90 PRINT ' "THIS IS A WRITING PAD." : PRINT" "
100 COLOUR 6
110 PRINT ' "THE OTHER WINDOW IS A DRAWING SHEET"
120 COLOUR 5
130 INPUT ' "DO YOU WANT TO DRAW";A $
140 IF A $= "N" THEN GOTO 170
150 MOVE 750,425 : GCOL0,RND (15) : DRAW 1100,425
160 PLOT 85,750,900
170 PRINT " " : PRINT " "
180 END
```

The output of the Program 7 will be two windows on the screen. After entering and RUNning the program. Type in CLS and press the RETURN key. You will see the text window is clear. Now type in CLG and press the RETURN key. You will see the graphics window is clear.

EXERCISES

1. Write a program to PRINT your name in double height on the screen. Add colour and also the flashing effect.
2. Prepare a program by using the VDU statement to PRINT the name of your school on the screen (see USER GUIDE for codes).

   Also use VDU 31,12,10 to set the TAB(12,10) so that the first alphabet of the name of your school will appear in the 12th column and 10th row.
3. What is the output of the following programs? Explain each of the statements.

   (i)
   ```
   10 MODE 5
   20 VDU 28,2,25,20,5 : COLOUR 130 : CLS : COLOUR 1
   30 PRINT "WHAT IS YOUR NAME";
   40 INPUT N $
   50 PRINT "HELLO";N $
   ```
   (ii)
   ```
   10 MODE 2
   20 VDU 24,400;200;1100;900;
   30 GCOL0,130 : CLG
   40 MOVE 400,20
   50 DRAW 1100,900
   60 MOVE 400,900
   70 DRAW 1100,200
   ```
4. Point out the error(s) in the following statements and correct them.
   ```
   (i) PRINT CHR (132) "INCOME TAX"
   (ii) VDU 24,800,150,1200,850
   (iii) VDU 24,1000;600;200;1000
   (iv) VDU 28;5;15;20;4
   (v) VDU 28,15,10,8,5
   (vi) PRINT CHR $ (130); CHR $ (136); CHR $ (141) "* DR. R.C.
        SHARMA *"
   ```
5. Which colour do you get by using the following function keys in MODE 7 along with shift key?
   (i) f1 (ii) f3 (iii) f7.
6. Which user-defined function key is used to produce the following effect in MODE 7 along with shift key?
   (i) flashing on, (ii) flashing off.
7. Which character code is used to get the double height text?
8. Modify Program 5 so that the word INDIA will appear in flashing red and cyan colours.

CHAPTER 19

Disc Handling

A user cannot save a program on a new disc until it is formatted.

Formatting The formatting of a disc is the process of preparing tracks and sectors on the new disc so that it can store any program in them. These tracks and sectors are the reference marks of memory location. It also creates a catalogue on the disc.

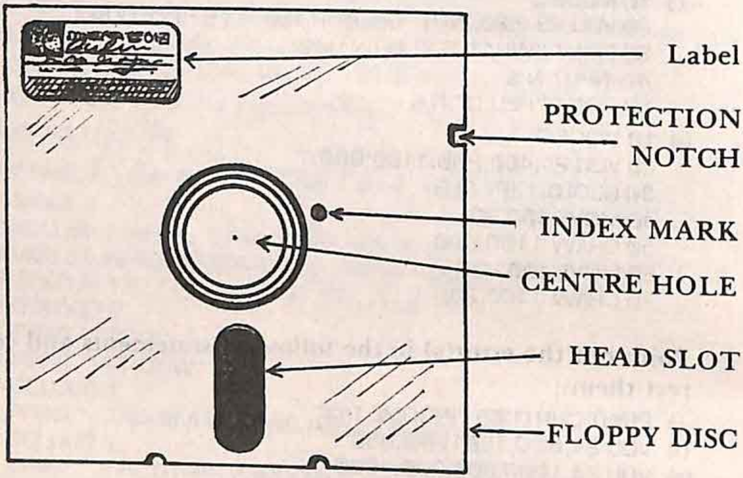


Fig. 19.1

Procedure for Formatting a Disc

1. Insert the UTILITY DISC.
2. Enter *FORM40 or *FORM80 for 40 track and 80 track disc respectively and press the RETURN key.
3. The following message will appear on the screen.
   Format which drive?
   Enter 0 and press the RETURN key.
   Another question will appear now
   Do you really want to format drive 0?
   (Do not reply at this point; just follow the next step (4).
4. Remove the UTILITY DISC.

5. Insert the disc which is to be formatted.
6. Then enter Y and press the RETURN key.

   You will see the formatting process going on. The number will appear on the screen.
7. Another question will appear when formatting is over. Disc formatted – – repeat (Y/N)?
8. If you have more discs to format then remove the disc from the disc drive and insert another which is to be formatted; then reply Y and press the RETURN key.

   Otherwise enter N and press the RETURN key.

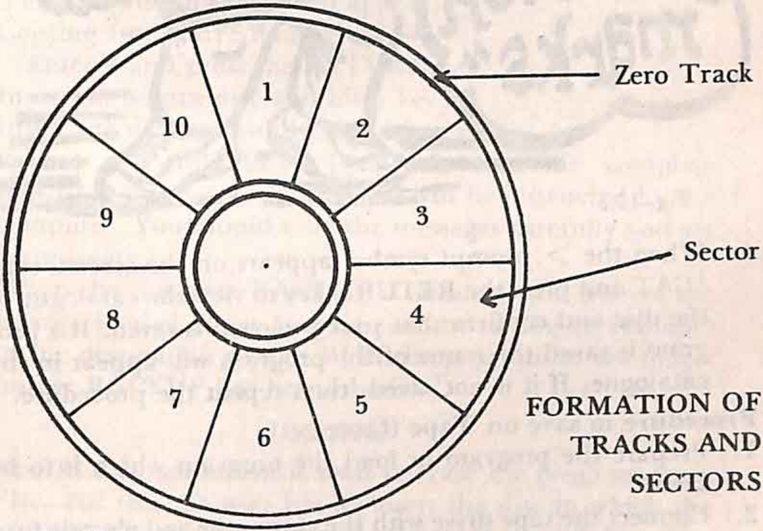


Fig. 19.2

PROGRAM SAVING

Saving Saving is the process of storing a program somewhere other than in the computer. The user can save a program on a magnetic tape or disc.

Procedure to Save on Disc

1. Prepare the program or load the program which is to be saved.
2. Insert the disc on which you want to save this program.
3. Type SAVE "name of the program" and press the RETURN key. The name of the program should have, at most, 10 characters.

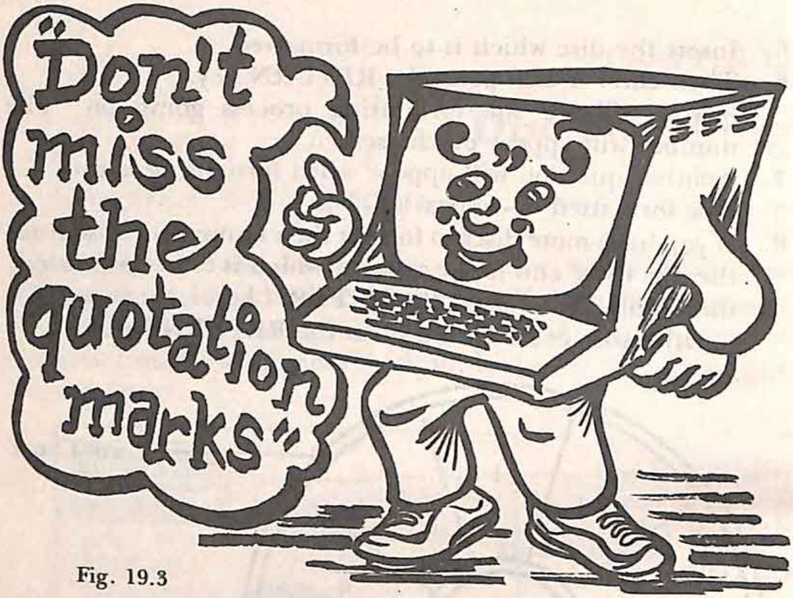

"Don't miss the quotation marks"

Fig. 19.3

4. When the > prompt symbol appears on the screen, type *CAT and press the RETURN key to view the catalogue of the disc and confirm that your program is saved. If a program is saved the name of the program will appear in the catalogue. If it is not saved, then repeat the procedure.

Procedure to save on Tape (Cassette)

1. Prepare the program or load the program which is to be saved.
2. Connect the tape drive with the computer and electric supply.
3. Set the tape in the position where you have to save your program
4. Type *TAPE and press the RETURN key
5. Type SAVE "name of the program" and press the RETURN key.
6. The message will appear "RECORD then RETURN"
   Press both the RECORD and PLAY keys of your tape recorder and then press the RETURN key.
7. You will see the tape moving. The prompt symbol > will appear when the program is saved.
8. Rewind the tape and load the program which was saved by you to see whether your program has been saved or not.

BACKUP

Sometimes it is necessary to have a duplicate copy of the software so that it can be used in case of failure of the original.

Procedures or equipment that are available for use in the event of failure or overloading of the normally used procedures or equipment are called 'BACKUPS'.

Procedure to Prepare a Backup

1. Type *ENABLE and press the RETURN key.
2. Type *BACKUP 0 0 and press the RETURN key.
3. The following message will appear:
   Copying from drive 0 to drive - -
   Enter 0 and press the RETURN key.
4. Insert the source disc and hit a key.
5. Insert the destination disc and hit a key.
   Repeat the procedures 4 and 5 until the complete BACKUP is done, for which you will be instructed by the computer. You should read the messages carefully and act accordingly.
6. When the complete BACKUP is obtained you will see the prompt symbol. Type *CAT to see the catalogue available in the destination disc or RUN the program to see that a proper BACKUP has been obtained.

DELETE

The DELETE statement is used to erase the program from the disc. For this the user has to insert the disc in which the particular program (which is to be deleted) is available. The format is as follows:

*DELETE "name of the program"

Let us presume, you want to erase a program named TRAINS from a disc. For this you have to type,

*DELETE "TRAINS"

Press RETURN key.

Now take a list of the catalogue and you will see that the program TRAINS is not in it.

CHAPTER 20

Sample Programs

In this chapter you are given some sample programs. You are advised to read the programs carefully, understand them completely and then try them on your microcomputer. It would be better still if you modify these programs and devise your own by using these programming ideas.

These programs have been provided merely as a guide for young users.

Program 1

```
 L.
 10 CLS: REM* TABLE OF N, N^2, N^3, 1/N*
 20 PRINT TABLE (8)" N", "      N^2", "      N^3", "      1/N"
 30 PRINT " "
 40 FOR I= 1 to 10
 50    N=I
 60    S=I^2
 70    C=I^3
 80    R= 1/I
 85    R= INT (R* 100)100
 90 PRINT N,S,C,R
100 NEXT I
110 END
```

Program 2

```
>L.
 10 CLS: PRINTTAB (12,2) "COMPOUND INTEREST"
 20 PRINTTAB (12,3) "= = = = = = = =  = = = = = = = ="
 30 INPUTTAB (2,4) "Enter Principal in Rs";P
 40 INPUTTAB (2,6) "Enter rate of interest in % ";R
 50 INPUTTAB (2,8) "Enter time in years";T
 60 PRINTTAB (2,10) "Select code number__________"
 70 PRINTTAB (2,11) "Compounded yearly __________1"
 80 PRINTTAB (2,12) "Compounded half yearly______2"
 90 PRINTTAB (2,13) "Compounded quarterly______ 4"
 100 INPUTTAB (2,16) "Enter code number__________";N
 105 LET R= R/N: LET T= T* N
 130 LET A= P*(1 + (R/100)) ^T
 140 LET A= INT (A* 100|100
 150 LET I= A - P
 160 PRINTTAB (2,20) "Amount ________Rs";INT (A* 10)/100
 170 PRINTTAB (2,22) "Compound interest- - Rs"; INT (I*100)/100
```

```
180 PRINTTAB (2,16) "Type RUN to restart"
190 END
```

Program 3

```
> L.
 10 CLS
 20 PRINTTAB (8,3) "F A C T O R I A L"
 30 INPUTTAB (5,6) "ENTER A POSITIVE INTEGER"; N
 40 IF N > 34 THEN PRINTTAB (5,8) "TOO BIG FOR ME": FOR wait= 1 TO
    2000: NEXT wait : PRINTTAB (31,6)" ":PRINTTAB (5,8)" " :GOTO30
 50 LET M= 1
 60 FOR K= 1 TO N
 70 M= M*K
 80 NEXT K
 90 PRINTTAB (3,15) "FACTORIAL OF ";N;"IS";M
100 END
```

Program 4

```
> L.
 10 CLS
 20 PRINT " "THIS PROGRAM GIVES YOU AN ORDER OF TWO
    NUMBERS."
 30 INPUT '"ENTER FIRST NUMBER";A
 40 INPUT '"ENTER SECOND NUMBER";B
 50 PRINT " " :PRINT" "
 60 IF A< B THEN GOTO 100: PRINT" "
 70 IF A> B THEN GOTO 120: PRINT " "
 80 PRINT '"BOTH NUMBERS ARE EQUAL."
 90 GOTO 130
100 PRINT A; "IS LESS THAN";B
110 GOTO 130
120 PRINT A;" IS GREATERTHAN";B
130 END
```

Program 5

```
> L.
 10 LET C=0
 20 CLS:PRINT '"THIS PROGRAM COUNTS THE WORDS IN THE GIVEN
    SENTENCE. ":PRINT" "
 30 INPUT "ENTER THE WORD WHICH IS TO BE COUNTED & PRESS THE
    RETURN KEY.";A$
 40 PRINT " " :PRINT " "
 50 PRINT '"ENTER THE WORDS & PRESS THE RETURN KEY AFTER
    EACH WORD AND DO NOT FORGET TO PUT FULL STOP AT THE END"
 60 INPUT B$
 70 IF B$= "." THEN 120
 80 IF B$=A$ THEN 100
 90 GOTO 60
```

```
100 LET C= C+ 1
110 GOTO 60
120 PRINT " " :PRINT " "
130 PRINT ' ' " THE WORD '";A $;"' OCCURS";C; "TIMES."
140 END
```

Program 6

```
> L.
10 CLS: PRINT ' "THIS PROGRAM REPLIES TO YOUR QUERY REGARDING
    THE NUMBER OF BOOKS ON THE SUBJECT IN YOUR SCHOOL LIBRARY."
 20 PRINT ' "ENTER—NAME OF THE SUBJECT OR E TO END THE
    PROGRAM —PRESS THE RETURN KEY."
 30 INPUT R $:LET K $= MID $ (R $, 1,3)
 40 READ S $, N,H,E,O: LET N $ MID $ (S $,1,3)
 50 IF S $= "ZZZ" THEN PRINT'' "THERE IS NOT A SINGLE BOOK ON"
    ;R$;" IN OUR LIBRARY.":RESTORE: GOTO20
 60 IF R $= "E" THEN 230
 70 IF K $ < > N $ THEN 40
 80 PRINT '"THE SUBJECT";S $;"ARE ";N; "IN THE LIBRARY."
 90 PRINT '"IN HINDI MEDIUM";H: PRINT "IN ENGLISH MEDIUM";
    E:PRINT" IN OTHER MEDIUM";0
100 RESTORE
110 GOTO 20
120 DATA PHYSICS, 250, 120, 120,10
130 DATA BIOLOGY, 300,150,90,60
140 DATA MATHEMATICS, 540,300,240,0
150 DATA CHEMISTRY, 302,140,160,2
160 DATA SPORTS,106,75,28,2
170 DATA HISTORY,182,155,15,12
180 DATA GEOGRAPHY,113,76,34,3
190 DATA ENGLISH, 280,0,280,0
200 DATA HINDI,200,200,0,0
210 DATA URDU,600,0,0,0
220 DATA ZZZ,0,0,0,0
230 END
```

Program 7

```
> L.
10 CLS
20 PRINT TAB (4,3) "ENTER A WORD OR SENTENCES"
25 PRINT TAB (4,4) "Do not use comma(s) "
30 PRINT TAB (4,5)"= = = = = = = = = ="
40 INPUT K $
50 PRINT TAB (4,7)"= = = = = = = = = = ="
60 L= LEN (K $)
70 FOR N= 1 TO L STEP 1
80 PRINT TAB (15) MID $ (K $,N,1):REM* SPLIT OFF 1 CHARACTER AT A
    TIME*
90 NEXT N
```

```
100 PRINT TAB (4)"= = = = = = = = = ="
110 PRINT " "PRINT" ":PRINT" "
120 INPUT TAB (2) "DO YOU LIKE TO SEE THE NEXT PROGRAMME
    (Y/N)";A$
130 IF A$= "Y" THEN GOTO 160
140 IF A$= "N" THEN PRINT "**GOOD BYE**": GOTO 270
150 PRINT "REPLY UNACCEPTED": GOTO 120
160 CLS: PRINT TAB (4,3) "ENTER A.WORD OR SENTENCES"
165 PRINT TAB (4,4) "Do not use comma(s) "
170 PRINT TAB (4,5) "= = = = = = = = = = ="
180 INPUT B$
190 L= LEN (B $) : REM *COUNTS CHARACTERS*
200 PRINT " ":PRINT" ":PRINT "= = = = = = = = = = ":PRINT "REVERSE OF
     ABOVE__________":PRINT " " PRINT " "
210 FOR N= L TO 1 STEP -1
220 PRINT MID$ (B$,N,1);" ";: REM* LAST CHARACTER FIRST*
230 NEXT N
240 PRINT " "
250 PRINT "= = = = = = = = = = = = = = = = = = = ="
260 PRINT "TYPE RUN TO RESTART"
270 END
```

Program 8

```
> L.
 10 MODE 1: COLOUR 131 : CLS : COLOUR 1
 20 PRINT TAB (10,2) "SIMULTANEOUS EQUATION"
 30 PRINTTAB (5,4) "THIS PROGRAM GIVES THE SOLUTION OF": PRINT"
    EQUATIONS A1X+ B1 Y+ C1 = 0 AND A2X+ B2Y*C2 = 0": PRINT" ":
    PRINT" "
 40 PRINT "ENTER COEFFICIENT OF X: A1 & A2 ARE";: INPUT A1,A2:
    PRINT " "
 50 PRINT "ENTER COEFFICIENT OF Y :B1 & B2 ARE"; : INPUT B1,B2:
    PRINT" "
 60 PRINT "ENTER CONSTANT :C1 & C2 ARE ";: INPUT C1, C2: PRINT
    " ": PRINT" "
 70 LET X= (C1*B2 - B1*C2)/(B1*A2 - A1*B2) : Y= (A1*C2 - C1*A2)/
    (B1*A2 - A1*B2)
 80 PRINT "X= "; X:PRINT " ":PRINT "Y= ";Y
 90 FOR Y= 1 TO 1000: NEXT
100 PRINT " " : PRINT" "
110 PRINT "WANT ANOTHER GO (Y/N)= ";
120 INPUT A$
130 IF A$= "Y" THEN 10
140 END
```

Program 9

```
> L.
 30 MODE 1
 40 GCOL 0,2
 50 COLOUR 129: CLS
 60 COLOUR 0
 70 PRINTTAB(9,3) "KENDRIYA VIDYALAYA"
 80 COLOUR 3
 90 PRINTTAB(11,5) "Mathematics Lesson"
100 COLOUR 2
110 PRINTTAB(11,7) "Geometrical Figures"
120 COLOUR 3
130 PRINTTAB(9,9) "Which figure do you want"
140 PRINTTAB(13,11) "Name of the figure"
150 PRINTTAB(13,13) "1 Acute "TAB(24,13)"2 Right"
160 PRINTTAB(13,15) "3 Obtuse" TAB(24,15) "4 Square"
170 PRINTTAB(9,17) "Enter the number (1-4)";
180 K=1
190 COLOUR 3
200 PRINTTAB (31,17) " "
210 INPUT TAB(31,17)N
220 IFN=K THEN GOSUB 290
230 IFN=K+1 THEN GOSUB 390
240 IFN=K+2 THEN GOSUB 500
250 IFN=K+3 THEN GOSUB 630
260 IFN> 4 THEN GOTO 200
270 GOTO 170
280 END
290 CLS: MOVE 500, 700: DRAW 800,700: DRAW 600,900
300 PLOT 85,500,700
310 PRINTTAB(10,14) "The triangle whose angles"
320 PRINTTAB (10,15) "are less than 90 degree is"
330 PRINTTAB(10,16) "known as an Acute angled"
340 PRINTTAB(10,17) "triangle"
350 PRINTTAB(17,11) "Acute"
360 PRINTTAB(8,29) "Press SPACE BAR To Continue"
370 REPEAT UNTIL GET $ = " ":CLS: GOTO 130
380 RETURN
390 CLS:PRINTTAB (10,14) "The triangle whose one"
400 PRINTTAB (10,15) "angle is 90 degree is known"
410 PRINTTAB(10,16) "as a Right angled triangle."
420 PRINTTAB(17,11) "Right"
430 MOVE 500,700
440 DRAW 800,700
450 DRAW 500,850
460 PLOT 85,500,700
470 PRINTTAB(8,29) "Press SPACE BAR To Continue"
480 REPEAT UNTIL GET $ = " ":CLS: GOTO 130
490 RETURN
500 CLS: PRINTTAB(11,14) "The triangle whose one"
510 PRINTTAB(11,15) "angle is greater than 90"
520 PRINTTAB (11,16) "deg. and less than 180 degree"
530 PRINTTAB(11,17) "is known as an Obtuse angled"
540 PRINTTAB(11,18) "triangle."
```

```
550 PRINTTAB(17,11) "Obtuse"
560 MOVE 500,700
570 DRAW 800,700
580 DRAW 450,900
590 PLOT 85,500,700
600 PRINTTAB(8,29) "Press SPACE BAR To Continue"
610 REPEAT UNTIL GET $= " ":CLS: GOTO 130
620 RETURN
630 CLS: PRINTTAB (11,14) "The quadrilateral whose"
640 PRINTTAB (11,15) "all sides are equal and"
650 PRINTTAB (11,16) "each angle is a right angle"
660 PRINTTAB (11,17) "is known as SQUARE."
670 PRINTTAB (16,11) "Square"
680 MOVE 500,700
690 DRAW 720,700
700 PLOT 85,720,910
710 DRAW 500,900
720 PLOT 85,500,700
730 PRINTTAB (8,29) "Press SPACE BAR To Continue"
740 REPEAT UNTIL GET $= " ": CLS:GOTO 130
750 CLG: GOTO 90
760 RETURN
770 END
```

Program 10

```
>L.
 10 REM* QUE AND ANS *
 20 CLS: M=20: REM* 8 QUE*
 30 DIM Q $ (M)
 40 T $= "CAPITAL CITIES": REM* TITLE OF TEST*
 50 F=0 :REM* FLAG*
 60 REM* READ IN QUE AND ANS*
 70 FOR N=1 TO M STEP 1
 80 READ Q $ (N)
 90 NEXT N
100 S=0 : REM* SET SCORE TO ZERO*
110 CLS: REM* CLEAR SCREEN*
120 PRINT TAB (12,2) "COUNTRY & CAPITAL": PRINT " " :PRINT " "
130 PRINTTAB (4,4) "WHEN EACH QUESTION APPEARS,KEY"
140 PRINT "IN YOUR ANSWER. THEN PRESS";
150 PRINT "RETURN KEY." : PRINT " " : PRINT " "
160 PRINT "IF YOUR ANS ARE WRONG, DO YOU";
170 PRINT "WANT TO KNOW CORRECT ANSWER?": PRINT" "
180 INPUT "ANSWER YES OR NO";K $
190 REM* IF ANS IS YES, FLAG F IS SET TO 1*
200 IF K $= "YES" THEN F=1: GOTO 240
210 IF K $= "NO" THEN F=0: GOTO 240
220 PRINT "REPLY UNACCEPTABLE" :GOTO 180
230 REM* PRESENT QUE AND ASK FOR ANS*
240 CLS: REM* CLEAR SCREEN*
```

```
250 FOR N=1 TO M STEP 2
260 PRINT ' "WHAT IS THE CAPITAL OF"; Q $ (N); : INPUT Q ; A$
270 IF A $=Q $ (N+1) THEN S=S+1: GOTO 310
280 IF F=0 THEN 310
290 IF F=1 THEN PRINT "WRONG":PRINT" "
300 PRINT "THE CORRECT ANSWER IS_" Q $(N+1); " • "
310 NEXT N
320 REM* PRESENT SCORE AND OBSERVATIONS*
330 CLS: REM* CLEAR SCREEN*
340 PRINTTAB(12) "RESULTS" : PRINT " " : PRINT " "
350 W= (M/2)-S:P=(S/ (S+W))*100:REM* PERCENTAGE SCORE*
360 W= (M/2]-S:REM* WRONG ANS*
370 PRINT "YOU ANSWERED";S;"CORRECTLY."
380 PRINT "YOU ANSWERED";W;" INCORRECTLY.";PRINT' "
390 PRINT "THIS REPRESENTS A SCORE OF";P;" PERCENT.";PRINT " "
400 IF P< =40 THEN PRINT "NOT TOO GOOD WAS IT?": GOTO 440
410 IF P > =41 AND P< 60 THEN PRINT "NOT BAD":"GOTO 440
420 IF P > =60 AND P< 80 THEN PRINT "GOOD SCORE":"GOTO 440
430 PRINT "ABSOLUTELY MAGNIFICENT"
440 PRINT " ":PRINT " ":PRINT " "
450 INPUT "DO YOU WANT TO TRY AGAIN (YES/NO)";K $
460 IF K $="YES" THEN 10
470 IF K $="NO" THEN PRINT "GOOD BYE": GOTO 570
480 PRINT "REPLY UNACCEPTABLE": GOTO 450
490 REM*********
500 REM* SUB ROUTINE.P.LINE*
510 PRINT "________________"
520 REM* QUE AND ANS PAIRS*
530 DATA FRANCE,PARIS,USSR,MOSCOW,JAPAN,TOKYO
540 DATA SWEDEN,STOCKHOLM,UK,LONDON,CSSR,PRAGUE
550 DATA CHINA,BEIJING,NEPAL, KATHMANDU,USA,WASHINGTON
560 DATA INDIA,DELHI
570 END
```

Program 11

```
>L.
 10 CLS
 20 MODE1:COLOUR 2:PRINT "** CHEMISTRY QUIZ FOR 9TH CLASS**"
 30 PRINT "  "
 40 PRINT "  "
 50 PRINT "= = = = = = = = = = = = = = = = = = = ="
 60 COLOUR 2
 70 LET C=0
 80 READ C $, T $
 90 IF C $="ZZZZ" THEN 290
100 PRINT "WHAT IS THE SYMBOL OF ";" "C $;
110 COLOUR 3
120 INPUT A $
130 IF A $=T $ THEN 230 ELSE SOUND 2,-15,10,5
140 PRINT
```

```
150 PRINT "NO. SORRY! THE SYMBOL OF "C $; " ";"IS";
160 COLOUR 1: PRINT $
170 PRINT " "
180 PRINT "Please wait for next question"
190 X $= INKEY $ (250)
200 COLOUR 2
210 PRINT " "
220 GOTO 80
230 PRINT " "
240 PRINT "YES! THAT'S RIGHT"
250 COLOUR 2
260 LET C= C+ 1
270 PRINT "                                        ";C
280 GOTO 80
290 PRINT "THAT'S THE END OF THE QUIZ"
300 PRINT "YOUR SCORE IS";C;" OUT OF 30"
310 LET X= (C/30)*100
320 PRINT "ACCURACY=" ;INT (X)"%"
330 REM*DATA*
340 DATA BARIUM, Ba
350 DATA CALCIUM, Ca
360 DATA CHLORINE, Cl
370 DATA ARGON, Ar
380 DATA ALUMINIUM, Al
390 DATA ZINC, Zn
400 DATA MERCURY, Hg
410 DATA SILVER, Ag
420 DATA GOLD, Au
430 DATA HELIUM, He
440 DATA CADMIUM, Cd
450 DATA COBALT, Co
460 DATA CHROMIUM, Cr
470 DATA CARBON, C
480 DATA SODIUM, Na
490 DATA POTASSIUM, K
500 DATA PHOSPHORUS, P
510 DATA SULPHUR, S
520 DATA OXYGEN, O
530 DATA BROMINE, Br
540 DATA NITROGEN, N
550 DATA MAGNESIUM, Mg
560 DATA MANGANESE, Mn
570 DATA NEON, Ne
580 DATA IODINE, I
590 DATA BISMUTH, Bi
600 DATA LEAD, Pb
610 DATA RADIUM, Ra
620 DATA RADON, Rn
630 DATA SILICON, Si
```

```
640 DATA ZZZZ, END OF FILE
```

You may add some more elements in the DATA of the above program.

Program 12

```
>L.
 10 MODE 2
 20 CLS : REM * DIGITAL CLOCK*
 30 S=0: M=0: H=0
 40 PRINTTAB(10,10); " "; TAB(13,10);" ";TAB(16,10)" "
 50 COLOUR 1
 60 PRINT TAB(7,10);H
 70 COLOUR 2
 80 PRINT TAB(10,10);M
 90 COLOUR 4
100 PRINT TAB(13,10);S
110 GCOL 0,5
120 MOVE400,600: DRAW 1000,600: DRAW 1000,780: DRAW 400,780:
    DRAW 400,600
130 S=S+1
140 X=INKEY(100)
150 IF S< >6 THEN 40
160 S=0: M=M+1
170 IF M< >60 THEN 40
180 M=0: H=H+1
190 IF H< >24 THEN 40
200 H=0
210 GOTO 60
220 END
```

Program 13

```
>L.
 10 MODE 2: COLOUR 133 : CLS
 20 COLOUR 135 : COLOUR 1 : PRINTTAB(1,1) "y": PRINTTAB(18,30)
    "x": PRINTTAB(1,30) "(0,0)": COLOUR 12
 30 PRINTTAB(4,4) "GRAPH Y=X/2"
 40 REM * AXIS *
 50 FOR N=20 TO 1200
 60 PLOT69,N,20
 70 NEXT N
 80 FOR M=30 TO 1000
 90 PLOT69,40,M
100 NEXT M
105 REM * Plotting the graph *
110 FOR X=20 TO 1200
120 Y=X/2
130 PLOT69,X,Y
140 NEXT X
150 END
```

Program 14

```
>L.
 10 CLS: R=0: W=0
 20 MODE 1: COLOUR 1
 30 PRINT TAB(15,6) "MATHS TEST": COLOUR 2
 40 X=RND(10): Y=RND(10): Z=RND(10)
 50 PRINT TAB(14,12) " ";X;"*";Y;"*"Z;"=";
 60 INPUT TAB(25,12),A
 70 IF A=X*Y*Z THEN 130
 80 FOR T=1 TO 3000: NEXT T
 90 PRINT TAB(14,12); "  WRONG  ":W=W+1: PRINT- TAB(20,3)
    "WRONG ANS =": W
100 FOR T=1 TO 3000: NEXT
110 PRINT TAB(14,12); "TRY AGAIN"
120 FOR T=1 TO 3000: NEXT T: GOTO 50
130 PRINT TAB(14,12)" CORRECT!          ":R=R+1: PRINT TAB(20,2)
    "RIGHT ANS=";R
140 FOR T=1 TO 3000: NEXT T
150 INPUT TAB(12,12) "ANOTHER QUESTION? (Y/N) ", Q$
160 FOR T=1 TO 2000: NEXT T
170 IF Q$="Y" THEN GOTO 20
180 PRINT TAB(12,12) "YOUR SCORE IS" ; INT((R* 100)/(R+W)) ; "%"
190 END
```

If you want to make it still more complicated, replace the line number 40,i.e. RND (10) by RND (20) or RND (30).

Program 15

```
>L
 10 MODE 2: COLOUR 12
 20 PRINTTAB(4,4) " sargam"
 30 FOR N=1 TO 2
 40 FOR P=1 TO 8
 50 READ A
 60 SOUND 1, -12,A,30
 70 NEXT P
 80 FOR W=1 TO 250: NEXT W
 90 FOR R=1 TO 8
100 READ B
110 SOUND 1, -15,B,15
120 NEXT R
130 FOR W=1 TO 300: NEXT W
140 RESTORE
150 NEXT N
160 DATA 56,64,72,76,84,92,100,104
170 DATA 104,100,92,84,76,72,64,56
180 END
```

Program 16

```
>L.
```

```
 10 MODE 2: COLOUR 134: CLS: COLOUR 5
 20 PRINTTAB (3,10) "MUSIC PROGRAM 1"
 30 COLOUR 4
 40 PRINTTAB (2,25) "PRESS ESCAPE TO     EXIT."
 45 REM *Please Refer USER GUIDE for ENVELOPE statement*
 50 ENVELOPE 1,1,0,0,0,0,0,0,127,-2,-10,-1,126,0
 60 REPEAT
 70 READ amplitude, pitch, duration
 80 IF amplitude=0, THEN RESTORE
 90 SOUND1,1, pitch, duration
100 UNTIL FALSE
110 DATA -15, 129,10,-15,117,10,-15,129,10
120 DATA -15,117,10,-15,101,10,-15,109,5
130 DATA -15,117,5,-15,121,10,-15,109,10
140 DATA -15,129,10,-15,117,10,-15,101,20,0,0,0
150 END
```

Line 45 can be omitted without affecting the output of the program.

Program 17

```
>L.
 10 MODE 2 : COLOUR 2
 20 CLS
 30 PRINTTAB (6,8) "SARANG"
 40 PRINTTAB (6,9) "========="
 50 FOR S=1 TO 2
 60 FOR P=1 TO 10
 70 READ F
 80 SOUND 1,-14,F,7
 90 NEXT P
100 RESTORE
110 NEXT S
120 DATA 133,141,153,161,177,181
130 DATA 181,177,161,153,141,133
140 FOR X=1 TO 3
150 FOR Q=1 TO 10
160 READ G
170 SOUND 1,-14,G,10
180 NEXT Q
190 RESTORE
200 NEXT X
210 RESTORE
220 FOR R=1 TO 10
230 READ H
240 SOUND 3,-10,H,10
250 NEXT R
260 RESTORE
270 FOR S=1 TO 10
280 READ M
```

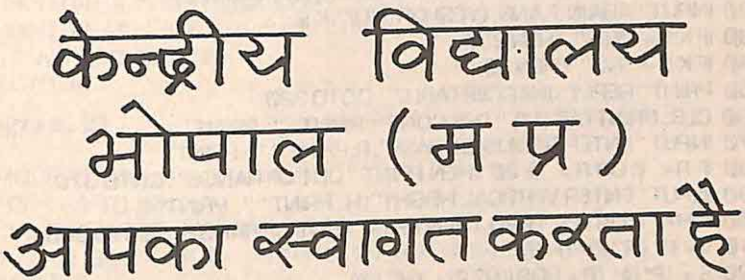
```
290 SOUND 1, - 14,M,15
300 NEXT S
310 RESTORE
320 FOR A = 1 TO 5
330 READ Z
340 SOUND 2, - 5,Z,20
350 NEXT A
360 DATA 133,141,153,161,177,181
370 DATA 181,177,161,153,141,133
380 END
```

Program 18

```
> L.
  10 MODE 2: COLOUR 3: REM*SWAGAT*
  20 GCOL 0,4
  30 VDU 23,225,255,8,63,73,73,59,8,8
  40 PRINT TAB(6,15) CHR$ (225): FOR W = 1 TO 500: NEXT: SOUND 1,
     - 15,133,10
  50 VDU 23,226,0,96,144,200,8,8,8,8
  60 PRINT TAB (6,14) CHR$ (226): FOR W = 1 TO 500: NEXT: SOUND 1,
     - 15,141,10
  70 VDU 23,227,255,0,0,127,80,32,0,0
  80 PRINT TAB (7,15) CHR$ (227) : FOR W = 1 TO 500: NEXT:
     SOUND 1, - 15,153,10
  90 VDU 23,228,255,9,121,129,137,125,25,40
 100 PRINTTAB (8,15) CHR$ (228): FOR W = 1 TO 500: NEXT:
     SOUND 1, - 15,161,10
 110 VDU 23,229,0,0,6,9,9,9,9,5
 120 PRINT TAB (8,14) CHR$ (229) : FOR W = 1 TO 500: NEXT:
     SOUND 1, - 15,177,10
 130 VDU 23,230,255,58,10,10,58,66,62,2
 140 PRINTTAB (9,15) CHR$ (230) : FOR W = 1 TO 500; NEXT: SOUND 1,
     - 15,181,10
 150 VDU 23,231,255,130,158,162,162,158,130,130
 160 PRINTTAB (12,15) CHR$ (231) : FOR W = 1 TO 500: NEXT: SOUND 1,
     - 15,133,10
 170 VDU 23,232,0,16,56,68,130,130,130,130
 180 PRINTTAB (12,14) CHR$ (232) : FOR W = 1 TO 500: NEXT: SOUND 1,
     - 15,141,10
 190 VDU 23,233,255,37,101,133,101,141,117,5
 200 PRINTTAB (13,15) CHR$ (233) : FOR W = 1 TO 500: NEXT: SOUND 1,
     - 15,153,10
 210 VDU 23,234,255,2,2,2,54,73,137,66
 220 PRINTTAB (14,15) CHR$ (234) : FOR W = 1 TO 500: NEXT: SOUND 1,
     - 15,161,10
 230 VDU 23,235,255,58,10,10,58,66,62,2
 240 PRINTTAB (15,15) CHR$ (235) : FOR W = 1 TO 500: NEXT: SOUND 1,
     - 15,177,10
 250 VDU 23,236,255,106,42,42,122,42,10,10
 260 PRINT TAB (7,18) CHR$ (236) : FOR W = 1 TO 500: NEXT: SOUND 1,
    -15,133,10
270 VDU 23,237,0,112,136,196,4,4,4,4
280 PRINT TAB(7,17) CHR$ (237) : FOR W= 1 TO 500: NEXT: SOUND 1,
    -15,133,10
290 VDU 23,238,255,37,37,37,29,5,5,5
300 PRINTTAB (8,18) CHR$ (238) : FOR W= 1 TO 50: NEXT: SOUND 1,
    -15,141,10
310 VDU 23,239,255,2,2,2,54,73,137,66
320 PRINTTAB (9,18) CHR$ (239) : FOR W= 1 TO 500: NEXT: SOUND 1,
    -15.153.10
330 VDU 23,240,248,224,224,224,224,224,224,248
340 PRINTTAB(11,18) CHR$ (240): FOR W= 1 TO 500: NEXT: SOUND 1.
    -15,161,10,
350 VDU 23,241,255,36,36,36,252,164,228,4
360 PRINTTAB (12,18) CHR$ (241); ".": FOR W= 1 TO 500: NEXT:
    SOUND 1,-15,177,10
370 VDU 23,242,255,34,34,34,30,6,10,18
380 PRINTTAB(13,18) CHR$ 242): FOR W= 1 TO 500: NEXT: SOUND 1,
    -15,181,10
390 VDU 23,243,31,7,7,7,7,7,7,31
400 PRINTTAB(14,18) CHR$ (243): FOR W= 1 TO 500: NEXT: SOUND 1,
    -15,181,10
410 VDU 23,244,255,229,21,37,93,37,21,229
420 PRINTTAB (4,20) CHR$ (244): FOR W= 1 TO 500: NEXT: SOUND 1,
    -15,177,10
430 VDU 23,245,255,36,36,36,28,4,4,4
440 PRINTTAB (5,20) CHR$ (245): FOR W= 1 TO 500: NEXT: SOUND 1,
    -15,161,10
450 VDU 23,246,255,17,253,149,149,245,17,17
460 PRINTTAB (6,20) CHR$ (246): FOR W= 1 TO 500: NEXT: SOUND 1,
    -15,153,10
470 VDU 23,247,255,8,8,47,48,32,40,24
480 PRINTTAB (8,20) CHR$ (247): FOR W= 1 TO 500: NEXT: SOUND 1,
    -15,141,10
490 VDU 23,248,255,10,250,138,138,250,10,10
500 PRINTTAB (9,20) CHR$ (248): FOR W= 1 TO 500: NEXT: SOUND 1,
    -15,133,10
510 VDU 23,249,255,36,36,36,100,36,4,4
520 PRINTTAB(10,20) CHR$ (249): FOR W= 1 TO 500: NEXT: SOUND 1,
    -15,133,10
530 VDU 23,250,255,8,248,136,136,136,136,72
540 PRINTTAB(11,20) CHR$ (250): FOR W= 1 TO 500: NEXT: SOUND 1,
    -15,141,10
550 VDU 23,251,255,8,63,73,73,59,8,8
560 PRINTTAB(13,20) CHR$ (251): FOR W= 1 TO 500: NEXT: SOUND 1,
    -15,153,10
570 VDU 23,252,255,8,8,40,48,32,40,24
580 PRINTTAB(14,20) CHR$ (252): FOR = 1 TO500: NEXT: SOUND 1,
    -15,161,10
```

```
590 VDU 23,253,255,10,250,138,138,138,138,74
600 PRINTTAB(15,20) CHR$ 253): FOR W= 1 TO 500: NEXT: SOUND 1,
    -15,177,10
610 VDU 23,254,255,8,48,64,48,80,68,120
620 PRINTTAB(17,20) CHR$ (254): FOR W= 1 TO 500: NEXT: SOUND 1,
    -15,181,10
630 VDU 23,255,0,0,54,82,18,18,20,8
640 PRINTTAB(17,19) CHR$ (255): FOR W= 1 TO 500: NEXT: SOUND 1,
    -15,181,10
650 MOVE 230,250: DRAW 1175,250: DRAW 1175, 600: DRAW
    230,600: DRAW 230,250
660 END
```

The output of the program 'SWAGAT' is



```
केन्द्रीय विद्यालय
भोपाल (म प्र)
आपका स्वागत करता है
```

Program 19

```
>L.
 10 CLS: P=3.141592654: REM * AREA AND VOLUME*
 20 PRINTTAB(10) "OPTION"
 25 PRINTTAB(1)" = = = = = ":PRINT" ":PRINT" "
 30 PRINT "CIRCLE ________________ 1": PRINT " "
 40 PRINT "SPHERE ________________ 2": PRINT " "
 41 PRINT "CONE __________________ 3": PRINT " "
 50 PRINT "CYLINDER ______________ 4": PRINT " ": PRINT " "
 60 INPUT "EXIT PROGRAM __________ 5": PRINT " ": PRINT " "
 70 INPUT "ENTER REQUIRED OPTION";N:PRINT" "
 80 IF N< 0 OR N > 5 THEN PRINT "OUT OF RANGE"; GOTO 70
 90 ON N GOTO 100,230,360,520,1000
100 CLS: PRINT TAB(10) "THE CIRCLE": PRINT" ": PRINT" "
110 INPUT "ENTER THE RADIUS";R:PRINT" "
120 IF R< 0 OR R>1E20 THEN PRINT "OUT OF RANGE" GOTO 110
130 C=2*P*R*:A=P*R*R
140 INPUT "HOW MANY DECIMAL PLACES";D: PRINT" ": PRINT" "
150 IF D< 0 OR D > 6 THEN PRINT " NEGATIVE VALUES/ TOO BIG FOR
     ME": GOTO 140
160 C=INT((10^D)*C+ (0.5))/ (10^D): A=INT((10^D) *A+ (0.5))/ (10^D)
170 PRINT "THE CIRCUMFERENCE IS";C; "UNITS": PRINT" "
180 PRINT "THE AREA IS ";A;"SQ. UNITS":PRINT" "
```

```
190 INPUT "AGAIN? ANS. (YES) OR(NO)"; K $
200 IF K $= "YES" THEN 100
210 IF K $= "NO" THEN 10
220 PRINT "UNACCEPTABLE REPLY": GOTO 190
230 CLS: PRINT TAB (10) "THE SPHERE": PRINT" ": PRINT" "
240 INPUT "ENTER RADIUS";R :PRINT" "
250 IF R< 0 OR R> 1E20 THEN PRINT "OUT OF RANGE" : GOTO 240
260 A=4*P*R*: V= (4/3)*P*R*R*R
270 INPUT "HOW MANY DECIMAL PLACES"; D: PRINT" ":PRINT" "
280 IF D< 0 OR D> 6 THEN PRINT "NEGATIVE VALUES/ TOO BIG FOR ME":
     GOTO 270
290 A= INT ((10^D)*A+ (0.5))/ (10^D): V= INT((10^D) *V+ (0.5))/ (10^D)
300 PRINT "THE SURFACE AREA IS ";A;" SQ. UNITS": PRINT" "
310 PRINT "THE VOLUME IS";V;"CUBIC UNITS": PRINT" "
320 INPUT "AGAIN ? ANS. (YES) OR(NO)";K $
330 IF K $= "YES" THEN 230
340 IF K $= "NO" THEN 10
350 PRINT "REPLY UNACCEPTABLE": GOTO 320
360 CLS: PRINTTAB (10) "THE CONE": PRINT" ": PRINT" "
370 INPUT "ENTER RADIUS OF BASE";R:PRINT " ": PRINT" "
380 IF R< 0 OR R> 1E20 THEN PRINT "OUT OF RANGE" :GOTO 370
390 INPUT "ENTER VERTICAL HEIGHT";H: PRINT" ": PRINT" "
400 IF H< 0 OR H> 1E20 THEN PRINT "OUT OF RANGE" : GOTO 390
410 V= P* (R^2)* (H/3)
420 S= (P*R)* (R+ SQR((R^2)+ (H^2)))
430 INPUT "HOW MANY DECIMAL PLACES";D: PRINT" ":PRINT" "
440 IF D< 0 OR D> 6 THEN PRINT "NEGATIVE VALUES/ TOO BIG FOR
     ME": GOTO 430
450 V= INT ((10^D)*V+ (0.5))/ (10^D): S= INT ((10^D)* S+ (0.5)/ (10^D)
460 PRINT "THE VOLUME IS";V; "CUBIC UNITS"; PRINT" "
470 INPUT "AGAIN ANS. (YES) OR(NO)";K $
480 IF K $= "YES" THEN GOTO 360
490 IF K $= "NO" THEN 10
500 PRINT "REPLY UNACCEPTABLE": GOTO 470
510 REM* GIVE GOOD QUES*
520 CLS: PRINT TAB (10) "THE CYLINDER": PRINT" ": PRINT" "
530 INPUT "ENTER RADIUS OF THE BASE";R : PRINT" "
540 IF R< 0.OR R> 1E20 THEN PRINT "OUT OF RANGE" : GOTO 530
550 INPUT "ENTER VERTICAL HEIGHT";H:PRINT" "
560 IF H< 0 OR H > 1E 20 THEN PRINT "OUT OF RANGE" : GOTO 550
570 V=P*R*R*H:S=2*P*R* (R+H)
580 INPUT "HOW MANY DECIMAL PLACES?":D: PRINT" "
590 IF D< 0 OR D > 6 THEN PRINT "NEGATIVE VALUES/ TOO BIG FOR
     ME": GOTO 580
600 V=INT ((10^D)*V+ (0.5)/(10^D): S= INT ))10^D)* S+
     (0.5)/(10^D)
610 PRINT "THE VOLUME OF THE CYLINDER IS ";V;" CUBIC UNITS":
     PRINT" "
620 PRINT "THE SURFACE AREA OF THE CYLINDER IS ";S;" SQ.
```

```
     UNITS": PRINT" "
 630 INPUT "AGAIN ANS. (YES) OR (NO)";K $
 640 IF K $= "YES" THEN 520
 650 IF K $= "NO" THEN 10
 660 PRINT "REPLY UNACCEPTABLE": GOTO 630
1000 PRINT "G O O D - B Y E"
1010 END
```

Program 20

```
> L.
10 REM * This is a computer kaleidoscope *
20 CLS
30 X= RND (18)
40 Y= RND (23)
50 C 128+ RND (6)
60 PRINTTAB (X,Y) CHR $ (C); CHR $ (255)
70 PRINTTAB (38 - X,Y) CHR $ (C); CHR $ (255)
80 FOR N= 1 TO 300: NEXT
90 GOTO 30
```

Program 21

```
> L.
10 MODE 2: REM* Random Circles *
15 FOR T= 1 TO 20
20 CX= RND (1200): CY= RND (1000)
30 R= RND (300): FOR X= - R TO R STEP 8: Y= SQR (R* R- X* X)
35 GCOL0,T
40 PLOT 69,CX+ X, CY+ Y
50 PLOT 69,CX+ X, CY- Y
60 NEXT X
70 NEXT T
```

Program 22

```
> L.
 10 REM This program will produce a display of ABABAB ... packing in
      crystals
 20 MODE 2
 30 GCOL0,1: FOR CX= 200 TO 900 STEP 100: CY= 600
 40 PROCC (CX)
 50 GCOL0,7: FORCX= 200 TO 900 STEP 100: CY= 500: PROCC (CX)
 60 GCOL0,2: FORCX= 200 TO 900 STEP 100: CY= 400: PROCC (CX)
 70 MOVE 100,300
 80 DRAW 100,700
 90 DRAW 1000,700
100 DRAW 1000,300
110 DRAW 100,300
120 END
130 DEFPROCC (CX)
140 R= 50: FOR X= - R TO R STEP 4: Y= SQR (R* R- X* X)
150 PLOT 69,CX+ X,CY+ Y
160 PLOT 69,CX+ X,CY- Y
```

```
170 NEXT X
180 NEXT CX
190 ENDPROC
```

Program 23

```
>L.
 10 MODE 7
 20 REM* TELETEXT CONTROL *
 30 PRINT CHR$ (134) "INDIA IS";CHR$ (129) "MY COUNTRY.":
 40 PRINT CHR$ (130) "ALL INDIANS ARE";CHR$ (134) "MY BROTHERS
    AND SISTERS.";
 50 PRINT CHR$ (131) "AND I AM PROUD OF";CHR$  (132) "ITS
    REACH";CHR$ (130) "VARIED HERITAGE";
 60 PRINT CHR$ (132) "I SHALL ALWAYS"; CHR$ (130) "TRY TO BE
    WORTHY OF IT.";
 70 PRINT CHR$ (133) "I SHALL GIVE RESPECT TO";
 80 PRINT CHR$ (130) "MY PARENTS";CHR$ (131) ",TEACHERS";CHR
    $ (133) "AND ALL ELDERS";
 90 PRINT CHR$ (134) "AND TREAT EVERY ONE WITH COURTESY.";
100 PRINT CHR$ (134) "TO MY PEOPLE AND MY COUNTRY I PLEDGE MY
     DEVOTION."; CHR$ (131)" IN THEIR WELL BEING AND PROSPERITY
     ALONE LIES MY" CHR$ (132) "HAPPINESS."
110 REM* No. 129 for red.*
120 REM* No. 130 for green.*
130 REM* No. 131 for yellow.*
140 REM* No. 132 for blue.*
150 REM* No. 133 for magenta.*
160 REM* No. 134 for cyan.*
170 REM* No. 135 for white*.
180 END
```

Program 24

```
>L.
  5 REM * This a computer design *
 10 MODE 2: COLOUR 140: CLS
 20 COLOUR 1
 30 MOVE 300,300
 40 GCOL0,9
 50 DRAW 300,600
 60 DRAW 800,600
 70 DRAW 800,300
 80 DRAW 300,300
 90 MOVE 350,550
100 DRAW 750,550
110 PLOT 85,550,450
120 DRAW 350,350
130 GCOL0,11
140 PLOT 85,750,350
150 DRAW 750,550
160 GCOL0,13
```

```
170 PLOT 85,550,450
180 DRAW 350,350
190 GCOL0,10
200 LOT 85,350,550
220 GCOL0,2
230 MOVE 250,250
240 DRAW 250,650
250 DRAW 850,650
260 DRAW 850,250
265 DRAW 250,250
270 PRINTTAB (1,4) "A computer design."
280 END
```

Program 25

```
> L.
 10 MODE 2: VDU5
 15 VDU 24,150;150;1100;800; : REM* GRAPHICS WINDOW*
 18 GCOL0,135: CLG
 20 FOR A= 1 TO 15
 30 C= RND (15)
 40 GCOL0,C
 50 R= 40+ RND (50)
 60 X= RND (1280)
 70 IF X<  R OR X> 1280– R THEN 60
 80 Y= RND (1024)
 90 IF Y<  R OR Y> 10 24 – R THEN 80
100 VDU29,X;Y;
110 FOR N= – R TO R STEP 5
120 M= SQR (R^2 – N^2)
130 MOVE N,M
140 DRAW N, – M
150 NEXT N
160 NEXT A
170 REM* Sweet sixteen Strawberries *
180 END
```

Program 26

```
> L.
 10 REM * MAXI, MINI and AVERAGE *
 2 C= 1
 3 SUM= 0
 40 READ M
 50 IF M= – 9999 THEN 170
 60 IF C> 1 THEN 90
 70 A= M
 80 T= M
 90 IF M > T THEN 130
100 IF M > = A THEN 140
110 A= M
120 GOTO 140
130 T= M
```

```
140 C=C+1
150 SUM=SUM+M
160 GOTO 40
170 PRINT '"MAXIMUM= ";T
180 PRINT '"MINIMUM= ";A
190 N=C-1
200 PRINT '"SUM"; SUM: PRINT'"OBSERVATIONS";N
210 PRINT '"AVERAGE= ";SUM/N
220 REM*DATA*
230 DATA 54,87,67,45,12,90,45
240 DATA 76,54,43,123,11
250 DATA 83,74,23,43,-9999
260 END
```

Program 27

```
>L.
  10 CLS
  20 PRINTTAB(5,2) CHR$(131);CHR$(136);CHR$(141); "INCOME TAX
     CONSULTANT"
  30 PRINTTAB(5,3) CHR$(131);CHR$(136);CHR$(141); "INCOME TAX
     CONSULTANT"
  40 INPUT '"YOUR NAME PLEASE";N$
  50 INPUT '"YOUR POST PLEASE";P$
  60 SUM=0:SD=10000
  70 PRINT'"STD DEDUCTION ALLOWED 30% OF THE THE SALARY OR Rs.
     10000 WHICHEVER IS LESS."
  80 INPUT '"HAVE YOU GOT A SUM OF YOUR SALARY p.a. (Y/N) ";R$
  90 PRINT '"**************************************************************"
 100 IF R$="Y" THEN INPUT '"YOUR TOTAL SALARY Rs"; SUM: GOTO
     130
 110 INPUT '' "ENTER SALARY FOR APRIL 87 TO MARCH 88 EXCLUDING
     HRA"; P1,P2,P3,P4,P5,P6, P7,P8,P9,P10,P11,P12
 120 SUM=P1+P2+P3+P4+P5+P6+P7+P8+P9+P10+P11+P12
 130 INPUT'"BONUS RECEIVED Rs."; B
 140 INPUT '"TOTAL arrears received Rs";A
 150 INPUT '"ANY OTHER INCOME Rs.";AI
 160 SUM=SUM+B+A+AI
 170 SDR=.3*(SUM)
 180 IF SDR< SD THEN SD=SDR
 190 IF SDR > SD THEN LET SD=10000
 200 PRINT '"DEDUCTIONS UNDER 80C"
 210 PRINT "= = = = = = = = = = = = = = = = = = = = = ="
 220 INPUT '"PF CONTRIBUTION p.a. Rs.";D1
 230 INPUT "TOTAL LIC PREMIUM PAID p.a. Rs";D2
 240 INPUT "NSC PURCHASED IN THIS YEAR Rs";D3
 250 D=D1+D2+D3
 260 IF D > 12000 THEN PD=9000+.4*(D-12000)
 270 IF D > 6000 THEN PD=6000+.5*(D-6000)
 280 IF D< =6000 THEN PD=D
 290 INPUT "DONATIONS PAID (80G) Rs";D4
```

```
300 INPUT "REBATE IN INCOME TAX(%) ";r
310 DR= (r/100)* (D4)
320 PRINT '"TOTAL INCOME  EXCLUDING HRA Rs"; SUM
330 PRINT "= = = = = = = = = = = = = = = =":PRINT" ": PRINT "STD
    DEDUCTIONS Rs";SD
340 PRINT "YOUR DEDUCTIONS Rs ";D
350 PRINT "Permissible deductions under 80C Rs";PD
360 PRINT "Permissible deductions under 80G Rs";DR
370 TI= SUM- (SD+ PD+ DR)
380 PRINT"________________________________________________":TI= IND(TI)
390 PRINT "'TAXABLE INCOME Rs";TI
400 IF TI< 18000 THEN TAX= 0: GOTO 440
410 IF TI< =25000 THEN TAX= .25* (TI- 18000): GOTO 440
420 IF TI< =50000 THEN TAX= 1750+ .3* (TI- 25000): GOTO 440
430 TAX= 9250+ .4* (TI- 50000)
440 PRINT'"YOUR INCOME TAX CALCULATED TO Rs";TAX
450 INPUT '"WHETHER YOU HAVE PAID SOME AMOUNT TOWARDS IN-
    COME TAX (Y/N)";A $
460 PAID= 0
470 IF A $= "Y" THEN INPUT "TOTAL SUM PAID BY YOU"; PAID 1:
    PAID= PAID+ PAID 1
480 TAX= INT(TAX- PAID)
490 PRINT: PRINT
500 PRINT "Mr/Mrs ";N $;" ";:PRINT'POST";P $
510 PRINT '"YOU HAVE TO PAY INCOME TAX Rs"; TAX
520 END
```

The output of the above program "INCOME TAX CONSULTANT" will be

```
INCOME TAX CONSULTANT

YOUR NAME PLEASE? AK VERMA
YOUR POST PLEASE? EDUCATION OFFICER
STD DEDUCTION ALLOWED 30% OF THE SALARY OR Rs. 10000
WHICHEVER IS LESS
HAVE YOU GOT SUM OF YOUR SALARY p.a. (Y/N)?Y
************************************************************************
YOUR TOTAL SALARY Rs? 48000
BONUS RECEIVED Rs? 1200
TOTAL arrears received Rs? 3050
ANY OTHER INCOME Rs? 3600
DEDUCTIONS UNDER 80C
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
PF CONTRIBUTION p.a. Rs.? 4500
TOTAL LIC PREMIUM PAID p.a. Rs? 2400
NSC PURCHASED IN THIS YEAR Rs? 4800
DONATIONS PAID (80G) Rs? 5000
REBATE IN INCOME TAX (%)? 50
TOTAL INCOME EXCLUDING HRA Rs 55850
```

```
========================================
STD DEDUCTIONS Rs 10000
YOUR DEDUCTIONS Rs 11700
Permissible deductions under 80C Rs 8850
Permissible deductions under 80G Rs 2500
----------------------------------------
TAXABLE INCOME Rs 34500
YOUR INCOME TAX CALCULATED TO Rs 4600
WHETHER YOU HAVE PAID SOME AMOUNT TOWARDS INCOME
  TAX (Y/N)?Y
TOTAL SUM PAID BY YOU? 1200
Mr/Mrs AK VERMA POST EDUCATION OFFICER
YOU HAVE TO PAY INCOME TAX Rs 3400
```

Program 28

```
>L.
10 MODE 1
20 MOVE 100,100
30 DRAW 100,1000
40 MOVE 900,100
50 DRAW 900,1000
55 GCOL0,1
60 FOR Y=100 TO 1000
70 PLOT 77,500,Y
80 NEXT
```

Program 29

```
10 REM* THIS PROGRAM PRODUCES THE SOUND OF SIRENS
20 FOR I=1 TO 10
30 SOUND 1,-15,96,11
40 SOUND 1,-15,84,11
50 NEXT
60 END
```

Program 30

```
>L.
 10 CLS: PRINTTAB(12,4) CHR$136;CHR$130 "SONG"
 20 FOR PLAY=1 TO 2
 30 FOR N=1 TO 32
 40 READ P,D
 50 SOUND 1,-15,P,D
 60 SOUND 1,0,0,0
 70 NEXT N
 80 RESTORE
 90 NEXT PLAY
100 DATA 52,8,60,8,72,16,72,16
110 DATA 88,8,100,8,108,16,108,16
120 DATA 108,8,100,8,100,8,92,16
130 DATA 92,8,88,16,88,16,92,16
140 DATA 88,8,72,8,72,16,60,16
150 DATA 80,8,72,8,72,16,60,16
```

```
160 DATA 72,8,88,4,80,4,72,4
170 DATA 80,8,80,16,72,16,72,16
180 DATA 00,00
190 END
```

Program 31

```
   L.
 10 MODE 2
 20 K= RAD (180/2000)
 30 M=0.7
 40 FOR X= -100 TO 100 STEP 2
 50 GCOL0 .RND(12)
 60 YM=6*INT(SQR (10000 - X*X) /5)
 70 FOR Y= YM TO -YM STEP -5
 80 LET Z=(SQR (X*X+Y*Y) )-M*Y
 90 IF Y=YM THEN 110
100 IF Z< Z1 THEN 120
110 PLOT 69,640 +4*X,500 + INT (2*Z)
120 Z1=Z
130 NEXT Y
140 NEXT X
150 FOR X= -100 TO 100 STEP 2
160 GCOL0, RND(12)
170 YM=6*INT(SQR (10000 -X*X) /5)
180 FOR Y= YM TO -YM STEP -5
190 LET Z=(SQR (X*X+Y*Y) ) -M*Y
200 IF Y=YM THEN 110
210 IF Z< Z1 THEN 120
220 PLOT 69,640-4*X,500 - INT (2*Z)
230 Z1=Z
240 NEXT Y
250 NEXT X
260 END
```

CHAPTER 21

Summary of Commands of the BBC Microcomputer

AND Allows a program to take a particular course of action only if both the conditions are met.

AUTO Automatically numbers program lines as a program is written; the first number and the spacing between the line numbers can be specified.

CHAIN Carries out the same function as LOAD + RUN. It transfers a program from a cassette tape into memory and then starts it RUNning.

CHR$ Converts a character code into a character.

CLG Clears the graphic area of the screen.

CLS Clears the text area of the screen.

COLOUR Selects the colour in which the text and its background can be shown on the screen.

COUNT Counts all the characters printed using PRINT, apart from comma, plus, semicolon, etc.

COS Trigonometric cosine function.

DATA The computer treats whatever follows DATA as information that may be needed later in the program; used in conjunction with READ.

DEG Converts a measurement in radians into degrees.

DELETE Deletes specified line(s) from a program.

DIV An operator which gives the whole number part of the division of two numbers, for example, 72/14 will result in 5.

DRAW A special case of the PLOT command. It draws a line in the current foreground colour from the graphics origin at 0,0 or from the last point visited, to X,Y.

END It marks the end of a program and prevents the computer carrying on into any subroutines that follow.

FOR ... NEXT A loop which repeats a sequence of program statements a specified number of times.

GCOL Selects a graphics colour.

GET Waits for a key to be pressed and then returns its ASCII number; GET$ returns a string.

GOSUB Makes the program jump to a subroutine beginning at the line number following the command.

GOTO Causes a program to jump to the line following the command.

IF ... THEN Prompts the computer to take a particular course of action only if the condition specified is detected.

INKEY Allows the user to wait for a specified time, during which the keyboard is monitored, and returns the ASCII number of any key pressed. INKEY$ returns the actual character pressed.

INPUT Instructs the computer to wait for some data from the keyboard which is then used in a program.

INT Converts a number with a decimal fraction into a whole number by rounding it off to the lower integer. For example, a statement A = INT(7.879) gives A = 7.

LEFT$ Forms a string from the left-hand part of given string.

LET Assigns a value to a variable. The use of LET is optional on the BBC Micro.

LEN Counts the number of characters in a string.

LIST Causes the computer to display a list of the program currently in its memory.

LOAD Transfers a program from a cassette tape or disc into the computer's memory.

MID$ Forms a string from the middle part of a given string.

MOD This function gives the remainder after division, such as 80/12 will give the number 8.

MODE Selects one of the BBC Micro's eight screen display modes.

MOVE A special case of the PLOT command. It moves the graphics cursor to the point specified without drawing anything on the screen.

NEW Removes a program from the computer's memory (the program can be recovered by using OLD).

OLD Recovers a program "DELETEd" from the computer's memory by typing NEW or pressing the BREAK key.

ON .. GOTO/GOSUB Makes the program jump to one of a number of statements or subroutines, depending on the value of a variable.

ON ERROR Directs the program to state where an error is detected.

OR Allows a program to take a particular course of action if either of two specified conditions is met.

PLOT Produces points, lines or triangles according to the option code that follows it. On the BBC Micro Model B 48 options are implemented. For example, PLOT 85 is used to shade the triangular area.

PRINT Causes whatever follows it to appear on the screen.

RAD Converts a measurement in degrees into radians.

READ Instructs the computer to obtain information from a DATA statement.

REM Enables the programmer to add remarks to a program. The computer ignores whatever follows REM in a program statement.

RENUMBER . Allows new line numbers to be given to a program. The first number and line spacing can be specified.

REPEAT ... UNTIL A loop which repeats a sequence of program statements until a specified condition is met.

RESTORE Resets the computer to READ the first item in a DATA list.

RETURN Terminates a subroutine (See also GOSUB).

RIGHT$ Forms a string from the right-hand part of a given string.

RND Produces numbers at random within specified limits.

SAVE Records a program currently in the computer's memory onto cassette tape or a disc. The program is identified by a file name.

SIN The trigonometric sine function.

SOUND Produces a range of sound specified by the numbers or variables following the command.

SQR Produces the positive square root of the number that follows it.

STEP Sets the step size in a FOR ... TO .. NEXT loop. It is optional to write STEP 1.

STOP Halts a program and PRINTS out the line number in which it appears.

TAB Sets the position of the text along a line.

VDU 23 Generates a character from a character code. VDU also has many other different uses.

CHAPTER 22

Computer Terminology

ALGORITHM A prescribed set of well-defined processing rules for the solution of a problem in a finite number of steps.

ALPHABETIC The set formed by the letters of the alphabet and special characters, such as period, comma, asterisk, etc.

ALPHANUMERIC A general term for alphabetic letters (A to Z), numerical digits (0 to 9), and special characters (– ,/,*, $,(,), + , etc; that are machine-processable.

ARITHMETIC LOGIC UNIT (ALU) That portion of the hardware of a computer that performs the mathematical and logical operations. It is a basic element of the central processing unit.

ANALYSIS Methodological investigation of a problem by any one of the consistent, systematic procedures.

APPLICATION PROGRAMS The programs, normally written by the user organization, that enables the computer to produce useful work; for example, inventory control, payroll, general ledger, class results processing, etc.

ARRAY

(1) A series of related items.
(2) An ordered arrangement or pattern of items or numbers, such as a determinant, matrix, vector or a table of numbers.

AMERICAN STANDARD CODE FOR INFORMATION INTERCHANGE (ASCII) Pronounced 'asskey'. A seven-bit standard code adopted by computers to represent characters. For example, in ASCII code, '2' is '0110010', 'A' is '1000001' and '$' is '0100100'.

ASSEMBLER A computer program that takes symbolic language instructions prepared by a computer user and converts them into computer readable instructions or machine language.

BACKUP Procedures or equipment that are available for use in the event of failure or overloading of the normally used procedures or equipment.

BEGINNER'S ALL-PURPOSE SYMBOLIC INSTRUCTION

CODE (BASIC) A higher-level computer programming language. BASIC has a small set of commands and simple statement formats. BASIC is thus widely used in programming instruction, in personal computing and in business and in industry.

BINARY The number system with a power of 2, or numbering system which uses only the digits 0 and 1.

BINARY ARITHMETIC A mathematical numeration system equivalent to our decimal arithmetic system but involving only two digits: 0 and 1.

BINARY DIGIT Either of the characters 0 or 1, abbreviated to bit.

BIT A binary digit; a digit (0 or 1) in the representation of a number in binary notation. Several bits make up a byte or a computer word.

BUG A term used to denote a mistake in a computer program or system,or a malfunction in a computer hardware component.

BYTE;

(1) A grouping of adjacent binary digits operated on by the computer as a unit. The commonest sized byte contains eight binary digits.

(2) A group of binary digits used to represent a single character.

CASSETTE A small plastic cartridge containing one-eighth inch magnetic tape for use in cassette recorders.

CASSETTE RECORDER A device designed to use cassettes to record and store digital data and, at a later time, reload this data into the computer's internal storage. It is used widely with microcomputers.

CATHODE RAY TUBE (CRT) An electronic tube with a screen upon which information may be displayed.

CENTRAL PROCESSING UNIT(CPU) The component of the computer system equipped with the circuitry to control the interpretation and execution of instructions.

CHARACTER Any symbol, digit, letter, or punctuation mark stored or processed by computing equipment.

CHIP A small, integrated circuit package containing many logical elements. A small piece of silicon impregnated with impurities in a pattern to form transistors, diodes, and resistors. Electrical paths are formed on it by depositing thin layers of aluminium or gold.

CIRCUIT A pathway designed for the controlled flow of electrons. A system of conductors and related electrical elements through which electrical currents flow. A communication link between two or more points.

COBOL (COMMON BUSINESS ORIENTED LANGUAGE) A high-level language developed for business data-processing applications.

COMMAND A control signal; a computer instruction.

COMMAND LANGUAGE A source language consisting primarily of commands capable of invoking a function.

COMPUTER An electronic device capable of solving problems or manipulating data by accepting data, performing prescribed operations (mathematical or logical) on the data, and supplying the results of these operations.

COMPUTER-ASSISTED INSTRUCTION (CAI) The use of the computer to augment individual instructions by providing the student with programmed sequences of instructions under computer control. The manner of sequencing and progressing through the materials permits students to progress at their own speed. CAI is a response to the individual need of the student.

COMPUTER LITERACY The non-technical study of the computer and its effect upon society. It is an important area in computer education as it provides the student with some of the knowledge, tools and understanding necessary to live in a society which is going to be dominated by computers.

COMPUTER PROGRAM The series of orderly statements and instructions that cause a computer to perform a particular operation or task.

COMPUTER PROGRAMMER A person skilled in the preparation of programs for a computer. A programmer designs, codes, debugs and documents computer programs.

COMPUTER SCIENCE The field of knowledge embracing all aspects of the design and use of computers.

COMPUTER SYSTEM The physical equipment and instructions, that is, hardware and software, used as a unit to process data. It includes the CPU, its operating system and the peripheral equipment such as printer, disc drive, keyboard, etc., and programs.

CONSOLE The part of the computer system that enables the operator to communicate with the system.

CONVERSATIONAL LANGUAGE A programming language that uses a near ENGLISH character set that facilitates communication between the user and the computer. BASIC is an example of a conversational language.

DATA A formalized representation of the facts or concepts suitable for communication, interpretation, or processing by people or by automatic means.

DATA BASE The collection of all data used and produced by a computer program. In large systems, data base analysis is usually concerned with large quantities of the data stored in disc and tape files. Smaller Microcomputer systems are more frequently concerned with data base allocations of available memory locations between the program and data storage areas, also known as data banks.

DATA PROCESSING:

1. One or more operations performed on data to produce the desired results.
2. The function of the computer centre.
3. A term used in reference to the operation performed by data processing equipment.

DEBUG To detect, locate and remove all mistakes in a computer program any malfunction in the computing system itself.

DISKETTE A floppy disc. A storage medium for microcomputers and minicomputers.

DISPLAY A visual representation of the data.

DISPLAY UNIT A device that provides a visual representation of the data. These are the cathode ray tube, the line printer.

DOCUMENT:

1. The medium and information recorded for human use.
2. A record that can be used by a person or a machine.

DOCUMENTATION The preparation of documents during system analysis and subsequent programming, that describe such things as the system, the program prepared, and the changes made at later dates.

DOT-MATRIX PRINTER A printing device in which the characters are formed by arranging the matrix of dots.

DRUM PRINTER A printing device that uses a drum embossed with alphanumeric characters.

EDIT:

1. To check the correctness of the data.
2. To change the form of data by adding or deleting certain characters, if necessary.

EDP (Electronic Data Processing) Data processing by electronic digital computers.

ERROR A general term referring to any deviation of a computer or of a measured quantity from the theoretically correct or true value.

EXECUTE To run a program on the computer or to carry out an instruction.

FILE A collection of related record treated as a unit.

FILE NAME A set of alphanumeric characters used to identify a particular file.

FLOPPY DISC A flexible disc coated with numeric substance that is stored in a paper or plastic envelope. It is used to store data.

FLOPPY DISC UNIT A peripheral storage device in which data are recorded or read using discs.

FLOW CHART A diagram that uses symbols and interconnected lines to show the logic and sequence of program operations.

FORMAT The arrangement of data on a VDU or paper.

FORTRAN A higher level language used to perform mathematical, scientific and engineering calculations.

GRAPHICS Facilities to provide computer output in the form of display drawings and pictures.

HARDWARE Physical equipment such as electronic, magnetic and mechanical devices.

HIGH LEVEL LANGUAGE A programming language oriented towards problem-solving.

INPUT The introduction of data from an external storage medium into a computer's internal unit.

INTERFACE A common boundary between two pieces of hardware or between two systems.

I/O DEVICES Abbreviation for INPUT and OUTPUT devices.

KILOBYTE A kilobyte is 1024 bytes. It is abbreviated as 'K'.

LANGUAGE A set of rules, representations and conventions used to communicate information (data).

LOAD To read information from the computer's storage.

LOGIC The science dealing with the formal principles of reasoning and thought.

MNEMONICS A group of letters or alphanumeric characters so grouped as to assist a programmer in remembering a word or phase.

MAGNETIC TAPE A plastic tape which has a magnetic surface for storing data.

MEMORY The storage facilities of the computer.

MENU A set of options listed on a computer's terminal.

MICROCOMPUTER A complete small computer system, consisting of hardware and software.

MICROPROCESSOR The central processing unit of a microcomputer. These are used as controlling devices.

MINICOMPUTER A digital computer that has a higher efficiency than microcomputers.

NUMERIC CONSTANT A constant pertaining to a number.

OFF-LINE Units of equipment that are not connected to the computer and hence not in direct communication with the computer.

ON-LINE Contrast with off-line.

OUTPUT:

1. Data transferred from a computer's internal storage or output device.
2. The final result given by the computer on paper, VDU, magnetic disc or tape.

OUTPUT DEVICE A unit that is used for taking out data from the computer and representing them in the desired form.

PASCAL A high level programming language.

PERSONAL COMPUTER A microcomputer used in the office or home or hobby club.

PROGRAM:

1. A sequence of instruction that permits a computer to perform a particular task.
2. To design write, and test one or more routines.

RANDOM ACCESS MEMORY (RAM) It is the main memory of a microcomputer.

READ ONLY MEMORY (ROM) Non erasable, permanently programmed memory.

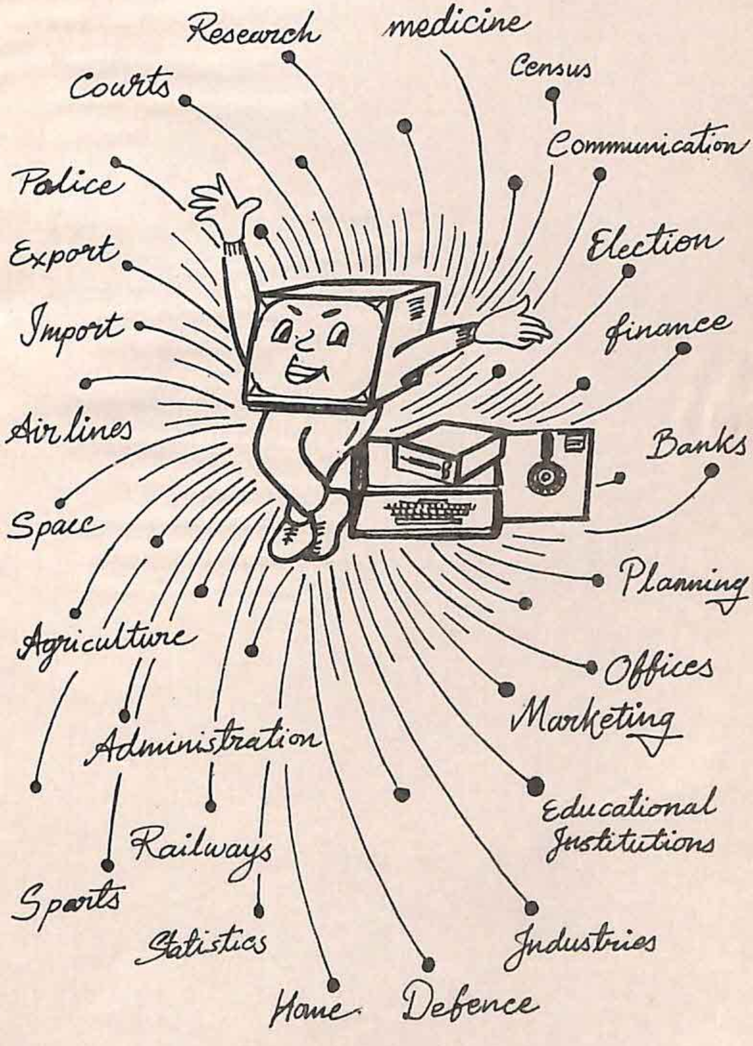
SOFTWARE A set of programs, procedures, routines and documents associated with computer operation.

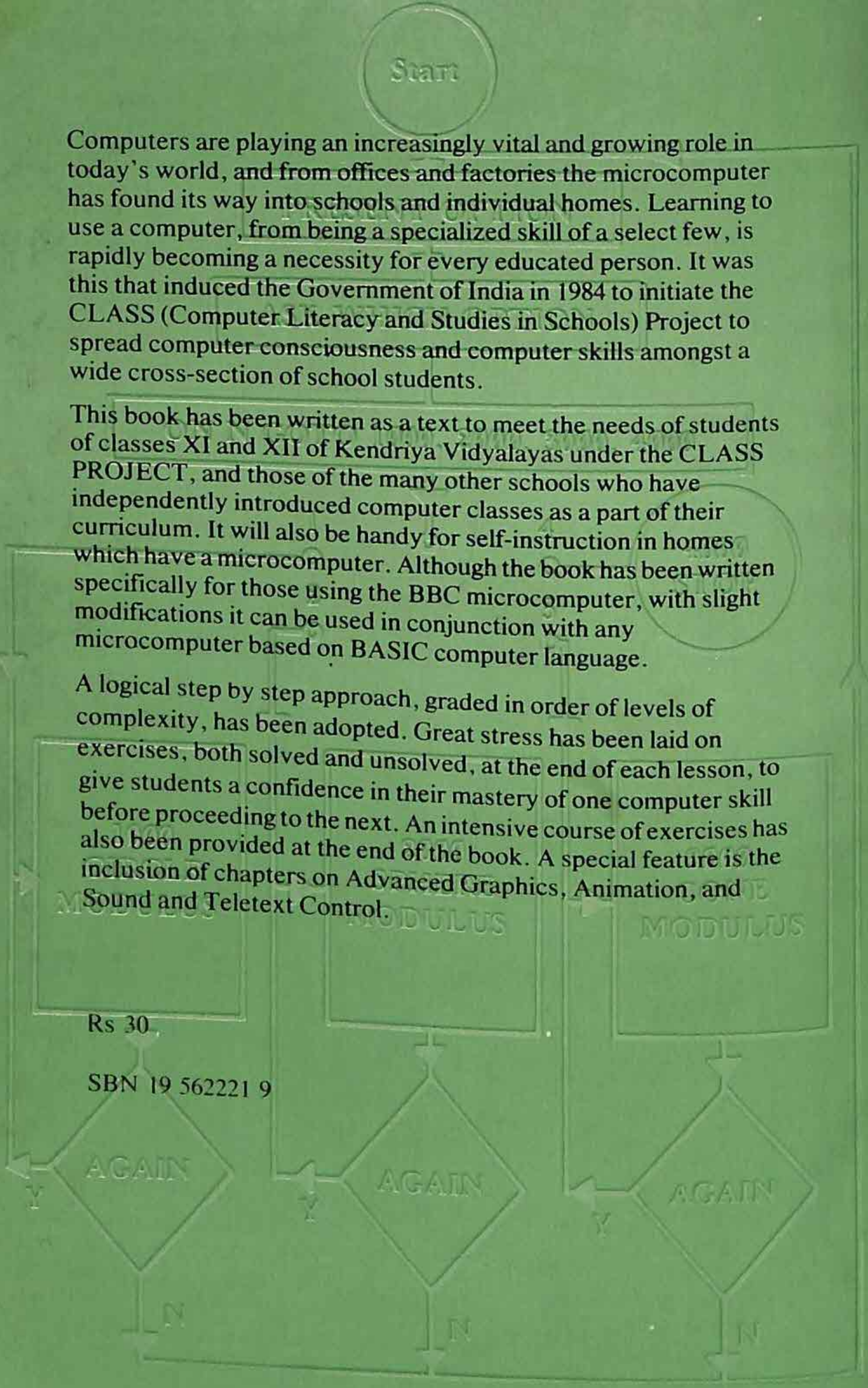
SOFTWARE PACKAGE A collection of computer programs along with printed material related to these programs.

STATEMENTS High level language expressions that define types of operations to be performed by the computer. A series of appropriate statements form a high level language program.

SYSTEM ANALYSIS A study of the activities, methods and techniques of organizational systems in order to determine what actions should be taken and how these actions can best be accomplished.

****





Computers are playing an increasingly vital and growing role in today's world, and from offices and factories the microcomputer has found its way into schools and individual homes. Learning to use a computer, from being a specialized skill of a select few, is rapidly becoming a necessity for every educated person. It was this that induced the Government of India in 1984 to initiate the CLASS (Computer Literacy and Studies in Schools) Project to spread computer consciousness and computer skills amongst a wide cross-section of school students.

This book has been written as a text to meet the needs of students of classes XI and XII of Kendriya Vidyalayas under the CLASS PROJECT, and those of the many other schools who have independently introduced computer classes as a part of their curriculum. It will also be handy for self-instruction in homes which have a microcomputer. Although the book has been written specifically for those using the BBC microcomputer, with slight modifications it can be used in conjunction with any microcomputer based on BASIC computer language.

A logical step by step approach, graded in order of levels of complexity, has been adopted. Great stress has been laid on exercises, both solved and unsolved, at the end of each lesson, to give students a confidence in their mastery of one computer skill before proceeding to the next. An intensive course of exercises has also been provided at the end of the book. A special feature is the inclusion of chapters on Advanced Graphics, Animation, and Sound and Teletext Control.

Rs 30

SBN 19 562221 9